

iPASOLINK EX
Advanced

OPEN SOURCE SOFTWARE



NEC Corporation

7-1, Shiba 5-Chome, Minato-Ku, Tokyo 108-8001, Japan

iPASOLINK EX Advanced

OPEN SOURCE SOFTWARE

This document contains licenses and notices for open source software used in this product. All intellectual property rights related to this product belong to NEC or the original right holder.

With respect to the open source software listed in this document, if you wish to receive a copy of any source code, please contact our sales representative.

Contact Form: <https://www.nec.com/contact/>

The target release version of this document is R3.1b

1. LICENSES 1-1

1.1	acl	1-1
1.2	apache2	1-17
1.3	apr-util	1-27
1.4	apr	1-36
1.5	attr	1-47
1.6	base-files	1-73
1.7	base-passwd	1-80
1.8	bash	1-87
1.9	beecrypt	1-100
1.10	bind	1-131
1.11	binutils	1-142
1.12	bridge-utils	1-182
1.13	bzip2	1-189
1.14	ca-certificates	1-190
1.15	chkconfig	1-198
1.16	coreutils	1-205
1.17	corosync	1-309
1.18	cpio	1-310
1.19	cracklib	1-323
1.20	cronie	1-333
1.21	curl	1-358

1.22	cyrus-sasl	1-359
1.23	db	1-360
1.24	dhcp	1-375
1.25	diffutils	1-376
1.26	dpkg	1-389
1.27	e2fsprogs	1-396
1.28	ebtables	1-455
1.29	elfutils	1-462
1.30	ethtool	1-475
1.31	eudev	1-568
1.32	expat	1-575
1.33	expect	1-576
1.34	file	1-577
1.35	findutils	1-578
1.36	flex	1-591
1.37	gawk	1-592
1.38	gdbm	1-605
1.39	gettext	1-618
1.40	glib-2.0	1-631
1.41	glibc	1-655
1.42	gmp	1-680
1.43	gnutls	1-702
1.44	gperftools	1-725
1.45	grep	1-726
1.46	gzip	1-739
1.47	hwdata	1-758
1.48	i2c-tools	1-765
1.49	ifupdown	1-772
1.50	inetutils	1-779
1.51	init-ifupdown	1-792

1.52	initscripts	1-793
1.53	iproute2	1-800
1.54	iptables	1-812
1.55	iputils	1-855
1.56	keyutils	1-928
1.57	kmod	1-933
1.58	krb5	1-943
1.59	less	1-969
1.60	libcap	1-982
1.61	libedit	1-990
1.62	libestr	1-991
1.63	libevent	1-1001
1.64	libfastjson	1-1003
1.65	libffi	1-1005
1.66	libgcc	1-1006
1.67	libgcrypt	1-1040
1.68	libgpg-error	1-1056
1.69	libgssglue	1-1096
1.70	libidn	1-1101
1.71	libnewt	1-1146
1.72	libnfsidmap	1-1155
1.73	libpam	1-1156
1.74	libpcap	1-1157
1.75	libpcre	1-1159
1.76	libproxy	1-1161
1.77	libtirpc	1-1173
1.78	libtool	1-1177
1.79	libunwind	1-1193
1.80	libusb-compat	1-1194
1.81	libusb1	1-1195

1.82	libxml2	1-1205
1.83	linux-windriver	1-1367
1.84	lldpd	1-1374
1.85	logrotate	1-1375
1.86	lzo	1-1382
1.87	m4	1-1393
1.88	MAKEDEV	1-1406
1.89	mingetty	1-1411
1.90	minicom	1-1418
1.91	mktemp	1-1431
1.92	mtd-utils	1-1432
1.93	ncurses	1-1443
1.94	neon	1-1444
1.95	net-snmp	1-1456
1.96	net-tools	1-1465
1.97	netbase	1-1494
1.98	netkit-rsh	1-1495
1.99	nettle	1-1502
1.100	nfs-utils	1-1529
1.101	ntp	1-1536
1.102	openais	1-1541
1.103	openssh	1-1542
1.104	openssl	1-1550
1.105	opkg-utils	1-1553
1.106	oprofile	1-1570
1.107	pam-radius-auth	1-1580
1.108	pciutils	1-1586
1.109	perl	1-1593
1.110	popt	1-1601
1.111	portmap	1-1602

1.112	ppp.....	1-1622
1.113	procps	1-1740
1.114	psmisc.....	1-1756
1.115	quagga.....	1-1763
1.116	radvd	1-1779
1.117	readline	1-1780
1.118	rpcbind	1-1793
1.119	rsync	1-1830
1.120	rsyslog.....	1-1843
1.121	run-postinsts	1-1861
1.122	screen	1-1862
1.123	sed	1-1881
1.124	setserial.....	1-1899
1.125	shadow	1-1904
1.126	shellinabox	1-1928
1.127	slang	1-1933
1.128	strace.....	1-1940
1.129	sudo.....	1-1941
1.130	sysklogd	1-2034
1.131	sysstat.....	1-2118
1.132	sysvinit	1-2125
1.133	tar.....	1-2132
1.134	tcl.....	1-2145
1.135	tcp-wrappers.....	1-2146
1.136	tcpdump	1-2147
1.137	time	1-2148
1.138	tnftp	1-2155
1.139	traceroute	1-2157
1.140	unzip	1-2173
1.141	update-rc.d	1-2175

1.142	usbutils	1-2180
1.143	util-linux	1-2187
1.144	vim-tiny	1-2204
1.145	vlan	1-2210
1.146	vsftpd.....	1-2220
1.147	watchdog-config.....	1-2227
1.148	watchdog.....	1-2228
1.149	which.....	1-2229
1.150	xinetd.....	1-2255
1.151	xz	1-2257
1.152	zip	1-2281
1.153	zlib	1-2283
1.154	zram-config.....	1-2316

1. LICENSES

1.1 acl

1.1.1 Available under license

Most components of the "acl" package are licensed under Version 2.1 of the GNU Lesser General Public License (see COPYING.LGPL).

Some components (as annotated in the source) are licensed under Version 2 of the GNU General Public License (see below),

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the

source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion

of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete

machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes

make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Most components of the "acl" package are licensed under Version 2.1 of the GNU Lesser General Public License (see below). below.

Some components (as annotated in the source) are licensed under Version 2 of the GNU General Public License (see COPYING).

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use

this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that,

in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever

changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library

facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed

through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR

CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.2 apache2

1.2.1 Available under license

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

excluding those notices that do not pertain to any part of the Derivative Works; and

- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be

liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

APACHE HTTP SERVER SUBCOMPONENTS:

The Apache HTTP Server includes a number of subcomponents with separate copyright notices and license terms. Your use of the source code for these subcomponents is subject to the terms and conditions of the following licenses.

For the mod_mime_magic component:

```
/*
 * mod_mime_magic: MIME type lookup via file magic numbers
 * Copyright (c) 1996-1997 Cisco Systems, Inc.
 *
 * This software was submitted by Cisco Systems to the Apache Group in July
 * 1997. Future revisions and derivatives of this source code must
 * acknowledge Cisco Systems as the original contributor of this module.
 * All other licensing and usage conditions are those of the Apache Group.
 *
 * Some of this code is derived from the free version of the file command
 * originally posted to comp.sources.unix. Copyright info for that program
 * is included below as required.
 * -----
 * - Copyright (c) Ian F. Darwin, 1987. Written by Ian F. Darwin.
 *
 * This software is not subject to any license of the American Telephone and
 * Telegraph Company or of the Regents of the University of California.
 *
 * Permission is granted to anyone to use this software for any purpose on any
 * computer system, and to alter it and redistribute it freely, subject to
 * the following restrictions:
 *
 * 1. The author is not responsible for the consequences of use of this
 * software, no matter how awful, even if they arise from flaws in it.
 *
 * 2. The origin of this software must not be misrepresented, either by
 * explicit claim or by omission. Since few users ever read sources, credits
 * must appear in the documentation.
 *
 * 3. Altered versions must be plainly marked as such, and must not be
 * misrepresented as being the original software. Since few users ever read
 * sources, credits must appear in the documentation.
 *
 * 4. This notice may not be removed or altered.
 * -----
 */
```

For the modules\mappers\mod_imagemap.c component:

"macmartinized" polygon code copyright 1992 by Eric Haines, erich@eye.com

For the server\util_md5.c component:

```
*****
 * NCSA HTTPd Server
 * Software Development Group
 * National Center for Supercomputing Applications
 * University of Illinois at Urbana-Champaign
 * 605 E. Springfield, Champaign, IL 61820
 * httpd@ncsa.uiuc.edu
```

```

*
* Copyright (C) 1995, Board of Trustees of the University of Illinois
*
*****
*
* md5.c: NCSA HTTPd code which uses the md5c.c RSA Code
*
* Original Code Copyright (C) 1994, Jeff Hostetler, Spyglass, Inc.
* Portions of Content-MD5 code Copyright (C) 1993, 1994 by Carnegie Mellon
* University (see Copyright below).
* Portions of Content-MD5 code Copyright (C) 1991 Bell Communications
* Research, Inc. (Bellcore) (see Copyright below).
* Portions extracted from mpack, John G. Myers - jgm+@cmu.edu
* Content-MD5 Code contributed by Martin Hamilton (martin@net.lut.ac.uk)
*
*/

/* these portions extracted from mpack, John G. Myers - jgm+@cmu.edu */
/* (C) Copyright 1993,1994 by Carnegie Mellon University
* All Rights Reserved.
*
* Permission to use, copy, modify, distribute, and sell this software
* and its documentation for any purpose is hereby granted without
* fee, provided that the above copyright notice appear in all copies
* and that both that copyright notice and this permission notice
* appear in supporting documentation, and that the name of Carnegie
* Mellon University not be used in advertising or publicity
* pertaining to distribution of the software without specific,
* written prior permission. Carnegie Mellon University makes no
* representations about the suitability of this software for any
* purpose. It is provided "as is" without express or implied
* warranty.
*
* CARNEGIE MELLON UNIVERSITY DISCLAIMS ALL WARRANTIES WITH REGARD TO
* THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY
* AND FITNESS, IN NO EVENT SHALL CARNEGIE MELLON UNIVERSITY BE LIABLE
* FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
* WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN
* AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING
* OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
* SOFTWARE.
*/

/*
* Copyright (c) 1991 Bell Communications Research, Inc. (Bellcore)
*
* Permission to use, copy, modify, and distribute this material
* for any purpose and without fee is hereby granted, provided
* that the above copyright notice and this permission notice
* appear in all copies, and that the name of Bellcore not be
* used in advertising or publicity pertaining to this
* material without the specific, prior written permission
* of an authorized representative of Bellcore. BELLCORE
* MAKES NO REPRESENTATIONS ABOUT THE ACCURACY OR SUITABILITY

```

* OF THIS MATERIAL FOR ANY PURPOSE. IT IS PROVIDED "AS IS",
* WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES.
*/

For the util_pcre.c and ap_regex.h components:

Copyright (c) 1997-2004 University of Cambridge

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
- * Neither the name of the University of Cambridge nor the names of its
contributors may be used to endorse or promote products derived from
this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

For the srclib\apr\include\apr_md5.h component:

/*
* This work is derived from material Copyright RSA Data Security, Inc.
*
* The RSA copyright statement and Licence for that original material is
* included below. This is followed by the Apache copyright statement and
* licence for the modifications made to that material.
*/

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.

License to copy and use this software is granted provided that it
is identified as the "RSA Data Security, Inc. MD5 Message-Digest
Algorithm" in all material mentioning or referencing this software
or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

*/

For the src/lib/apr/passwd/apr_md5.c component:

/*

* This work is derived from material Copyright RSA Data Security, Inc.

*

* The RSA copyright statement and Licence for that original material is included below. This is followed by the Apache copyright statement and licence for the modifications made to that material.

*/

/* MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm

*/

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

*/

/*

* The apr_md5_encode() routine uses much code obtained from the FreeBSD 3.0

* MD5 crypt() function, which is licenced as follows:

* -----

* "THE BEER-WARE LICENSE" (Revision 42):

* <phk@login.dknet.dk> wrote this file. As long as you retain this notice you

```
* can do whatever you want with this stuff. If we meet some day, and you think
* this stuff is worth it, you can buy me a beer in return. Poul-Henning Kamp
* -----
*/
```

For the srclib\apr-util\crypto\apr_md4.c component:

```
* This is derived from material copyright RSA Data Security, Inc.
* Their notice is reproduced below in its entirety.
*
* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
* rights reserved.
*
* License to copy and use this software is granted provided that it
* is identified as the "RSA Data Security, Inc. MD4 Message-Digest
* Algorithm" in all material mentioning or referencing this software
* or this function.
*
* License is also granted to make and use derivative works provided
* that such works are identified as "derived from the RSA Data
* Security, Inc. MD4 Message-Digest Algorithm" in all material
* mentioning or referencing the derived work.
*
* RSA Data Security, Inc. makes no representations concerning either
* the merchantability of this software or the suitability of this
* software for any particular purpose. It is provided "as is"
* without express or implied warranty of any kind.
*
* These notices must be retained in any copies of any part of this
* documentation and/or software.
*/
```

For the srclib\apr-util\include\apr_md4.h component:

```
*
* This is derived from material copyright RSA Data Security, Inc.
* Their notice is reproduced below in its entirety.
*
* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
* rights reserved.
*
* License to copy and use this software is granted provided that it
* is identified as the "RSA Data Security, Inc. MD4 Message-Digest
* Algorithm" in all material mentioning or referencing this software
* or this function.
*
* License is also granted to make and use derivative works provided
* that such works are identified as "derived from the RSA Data
* Security, Inc. MD4 Message-Digest Algorithm" in all material
* mentioning or referencing the derived work.
*
* RSA Data Security, Inc. makes no representations concerning either
* the merchantability of this software or the suitability of this
* software for any particular purpose. It is provided "as is"
```

```

* without express or implied warranty of any kind.
*
* These notices must be retained in any copies of any part of this
* documentation and/or software.
*/

```

For the src/lib/apr-util/test/testmd4.c component:

```

*
* This is derived from material copyright RSA Data Security, Inc.
* Their notice is reproduced below in its entirety.
*
* Copyright (C) 1990-2, RSA Data Security, Inc. Created 1990. All
* rights reserved.
*
* RSA Data Security, Inc. makes no representations concerning either
* the merchantability of this software or the suitability of this
* software for any particular purpose. It is provided "as is"
* without express or implied warranty of any kind.
*
* These notices must be retained in any copies of any part of this
* documentation and/or software.
*/

```

For the test/vb.c component:

```

/*          ZeusBench V1.01
=====

```

This program is Copyright (C) Zeus Technology Limited 1996.

This program may be used and copied freely providing this copyright notice is not removed.

This software is provided "as is" and any express or implied warranties, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall Zeus Technology Ltd. be liable for any direct, indirect, incidental, special, exemplary, or consequential damaged (including, but not limited to, procurement of substitute good or services; loss of use, data, or profits; or business interruption) however caused and on theory of liability. Whether in contract, strict liability or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

Written by Adam Twiss (adam@zeus.co.uk). March 1996

Thanks to the following people for their input:

Mike Belshe (mbelshe@netscape.com)

Michael Campanella (campanella@stevms.enet.dec.com)

```

*/

```

1.3 apr-util

1.3.1 Available under license

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

excluding those notices that do not pertain to any part of the Derivative Works; and

- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be

liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

APACHE PORTABLE RUNTIME SUBCOMPONENTS:

The Apache Portable Runtime includes a number of subcomponents with separate copyright notices and license terms. Your use of the source code for these subcomponents is subject to the terms and conditions of the following licenses.

For the include\apr_md5.h component:

```
/*
 * This is work is derived from material Copyright RSA Data Security, Inc.
 *
 * The RSA copyright statement and Licence for that original material is
 * included below. This is followed by the Apache copyright statement and
 * licence for the modifications made to that material.
 */
```

```
/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.
```

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

```
*/
```

For the passwd\apr_md5.c component:

```
/*
 * This is work is derived from material Copyright RSA Data Security, Inc.
 *
 * The RSA copyright statement and Licence for that original material is
 * included below. This is followed by the Apache copyright statement and
 * licence for the modifications made to that material.
 */
```

```
/* MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm
*/
```

```
/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.
```

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data

Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

```
*/
/*
* The apr_md5_encode() routine uses much code obtained from the FreeBSD 3.0
* MD5 crypt() function, which is licenced as follows:
* -----
* "THE BEER-WARE LICENSE" (Revision 42):
* <phk@login.dknet.dk> wrote this file. As long as you retain this notice you
* can do whatever you want with this stuff. If we meet some day, and you think
* this stuff is worth it, you can buy me a beer in return. Poul-Henning Kamp
* -----
*/
```

For the crypto\apr_md4.c component:

```
* This is derived from material copyright RSA Data Security, Inc.
* Their notice is reproduced below in its entirety.
*
* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
* rights reserved.
*
* License to copy and use this software is granted provided that it
* is identified as the "RSA Data Security, Inc. MD4 Message-Digest
* Algorithm" in all material mentioning or referencing this software
* or this function.
*
* License is also granted to make and use derivative works provided
* that such works are identified as "derived from the RSA Data
* Security, Inc. MD4 Message-Digest Algorithm" in all material
* mentioning or referencing the derived work.
*
* RSA Data Security, Inc. makes no representations concerning either
* the merchantability of this software or the suitability of this
* software for any particular purpose. It is provided "as is"
* without express or implied warranty of any kind.
*
* These notices must be retained in any copies of any part of this
* documentation and/or software.
*/
```

For the crypto\crypt_blowfish.c(h) component:

```
* Written by Solar Designer <solar at openwall.com> in 1998-2011.
* No copyright is claimed, and the software is hereby placed in the public
* domain. In case this attempt to disclaim copyright and place the software
* in the public domain is deemed null and void, then the software is
```

- * Copyright (c) 1998-2011 Solar Designer and it is hereby released to the
- * general public under the following terms:
- *
- * Redistribution and use in source and binary forms, with or without
- * modification, are permitted.
- *
- * There's ABSOLUTELY NO WARRANTY, express or implied.

See crypto/crypt_blowfish.c for more information.

For the include\apr_md4.h component:

- *
- * This is derived from material copyright RSA Data Security, Inc.
- * Their notice is reproduced below in its entirety.
- *
- * Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
- * rights reserved.
- *
- * License to copy and use this software is granted provided that it
- * is identified as the "RSA Data Security, Inc. MD4 Message-Digest
- * Algorithm" in all material mentioning or referencing this software
- * or this function.
- *
- * License is also granted to make and use derivative works provided
- * that such works are identified as "derived from the RSA Data
- * Security, Inc. MD4 Message-Digest Algorithm" in all material
- * mentioning or referencing the derived work.
- *
- * RSA Data Security, Inc. makes no representations concerning either
- * the merchantability of this software or the suitability of this
- * software for any particular purpose. It is provided "as is"
- * without express or implied warranty of any kind.
- *
- * These notices must be retained in any copies of any part of this
- * documentation and/or software.
- */

For the test\testmd4.c component:

- *
- * This is derived from material copyright RSA Data Security, Inc.
- * Their notice is reproduced below in its entirety.
- *
- * Copyright (C) 1990-2, RSA Data Security, Inc. Created 1990. All
- * rights reserved.
- *
- * RSA Data Security, Inc. makes no representations concerning either
- * the merchantability of this software or the suitability of this
- * software for any particular purpose. It is provided "as is"
- * without express or implied warranty of any kind.
- *
- * These notices must be retained in any copies of any part of this
- * documentation and/or software.

```
*/
```

For the xml\expat\conf\tools\install-sh component:

```
#
# install - install a program, script, or datafile
# This comes from X11R5 (mit/util/scripts/install.sh).
#
# Copyright 1991 by the Massachusetts Institute of Technology
#
# Permission to use, copy, modify, distribute, and sell this software and its
# documentation for any purpose is hereby granted without fee, provided that
# the above copyright notice appear in all copies and that both that
# copyright notice and this permission notice appear in supporting
# documentation, and that the name of M.I.T. not be used in advertising or
# publicity pertaining to distribution of the software without specific,
# written prior permission. M.I.T. makes no representations about the
# suitability of this software for any purpose. It is provided "as is"
# without express or implied warranty.
#
```

For the expat xml parser component:

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd
and Clark Cooper

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

=====

For the ldap/apr_ldap_url.c component:

```
/* Portions Copyright 1998-2002 The OpenLDAP Foundation
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted only as authorized by the OpenLDAP
 * Public License. A copy of this license is available at
```

```
* http://www.OpenLDAP.org/license.html or in file LICENSE in the
* top-level directory of the distribution.
*
* OpenLDAP is a registered trademark of the OpenLDAP Foundation.
*
* Individual files and/or contributed packages may be copyright by
* other parties and subject to additional restrictions.
*
* This work is derived from the University of Michigan LDAP v3.3
* distribution. Information concerning this software is available
* at: http://www.umich.edu/~dirsvcs/ldap/
*
* This work also contains materials derived from public sources.
*
* Additional information about OpenLDAP can be obtained at:
*   http://www.openldap.org/
*/

/*
* Portions Copyright (c) 1992-1996 Regents of the University of Michigan.
* All rights reserved.
*
* Redistribution and use in source and binary forms are permitted
* provided that this notice is preserved and that due credit is given
* to the University of Michigan at Ann Arbor. The name of the University
* may not be used to endorse or promote products derived from this
* software without specific prior written permission. This software
* is provided ``as is" without express or implied warranty.
*/
```

1.4 apr

1.4.1 Available under license

```

/* Licensed to the Apache Software Foundation (ASF) under one or more
 * contributor license agreements. See the NOTICE file distributed with
 * this work for additional information regarding copyright ownership.
 * The ASF licenses this file to You under the Apache License, Version 2.0
 * (the "License"); you may not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

#ifndef APR_LIB_H
#define APR_LIB_H

/**
 * @file apr_lib.h
 * This is collection of oddballs that didn't fit anywhere else,
 * and might move to more appropriate headers with the release
 * of APR 1.0.
 * @brief APR general purpose library routines
 */

#include "apr.h"
#include "apr_errno.h"

#if APR_HAVE_CTYPE_H
#include <ctype.h>
#endif
#if APR_HAVE_STDARG_H
#include <stdarg.h>
#endif

#ifdef __cplusplus
extern "C" {
#endif /* __cplusplus */

/**
 * @defgroup apr_lib General Purpose Library Routines
 * @ingroup APR
 * This is collection of oddballs that didn't fit anywhere else,
 * and might move to more appropriate headers with the release
 * of APR 1.0.
 * @{

```



```

*/

/** A constant representing a 'large' string. */
#define HUGE_STRING_LEN 8192

/*
 * Define the structures used by the APR general-purpose library.
 */

/** @see apr_vformatter_buff_t */
typedef struct apr_vformatter_buff_t apr_vformatter_buff_t;

/**
 * Structure used by the variable-formatter routines.
 */
struct apr_vformatter_buff_t {
    /** The current position */
    char *curpos;
    /** The end position of the format string */
    char *endpos;
};

/**
 * return the final element of the pathname
 * @param pathname The path to get the final element of
 * @return the final element of the path
 * @remark
 * <PRE>
 * For example:
 *      "/foo/bar/gum"  -> "gum"
 *      "/foo/bar/gum/" -> ""
 *      "gum"           -> "gum"
 *      "bs\\path\\stuff" -> "stuff"
 * </PRE>
 */
APR_DECLARE(const char *) apr_filepath_name_get(const char *pathname);

/**
 * apr_killpg
 * Small utility macros to make things easier to read. Not usually a
 * goal, to be sure..
 */

#ifdef WIN32
#define apr_killpg(x, y)
#else /* WIN32 */
#ifdef NO_KILLPG
#define apr_killpg(x, y) (kill(-(x), (y)))
#else /* NO_KILLPG */
#define apr_killpg(x, y) (killpg((x), (y)))
#endif /* NO_KILLPG */
#endif /* WIN32 */

/**

```

```

* apr_vformatter() is a generic printf-style formatting routine
* with some extensions.
* @param flush_func The function to call when the buffer is full
* @param c The buffer to write to
* @param fmt The format string
* @param ap The arguments to use to fill out the format string.
*
* @remark
* <PRE>
* The extensions are:
*
* - %%pA takes a struct in_addr *, and prints it as a.b.c.d
* - %%pI takes an apr_sockaddr_t * and prints it as a.b.c.d:port or
* \[ipv6-address\]:port
* - %%pT takes an apr_os_thread_t * and prints it in decimal
* ('0' is printed if !APR_HAS_THREADS)
* - %%pt takes an apr_os_thread_t * and prints it in hexadecimal
* ('0' is printed if !APR_HAS_THREADS)
* - %%pm takes an apr_status_t * and prints the appropriate error
* string (from apr_strerror) corresponding to that error code.
* - %%pp takes a void * and outputs it in hex
* - %%pB takes a apr_uint32_t * as bytes and outputs it's apr_strsize
* - %%pF same as above, but takes a apr_off_t *
* - %%pS same as above, but takes a apr_size_t *
*
* %%pA, %%pI, %%pT, %%pp are available from APR 1.0.0 onwards (and in 0.9.x).
* %%pt is only available from APR 1.2.0 onwards.
* %%pm, %%pB, %%pF and %%pS are only available from APR 1.3.0 onwards.
*
* The %%p hacks are to force gcc's printf warning code to skip
* over a pointer argument without complaining. This does
* mean that the ANSI-style %%p (output a void * in hex format) won't
* work as expected at all, but that seems to be a fair trade-off
* for the increased robustness of having printf-warnings work.
*
* Additionally, apr_vformatter allows for arbitrary output methods
* using the apr_vformatter_buff and flush_func.
*
* The apr_vformatter_buff has two elements curpos and endpos.
* curpos is where apr_vformatter will write the next byte of output.
* It proceeds writing output to curpos, and updating curpos, until
* either the end of output is reached, or curpos == endpos (i.e. the
* buffer is full).
*
* If the end of output is reached, apr_vformatter returns the
* number of bytes written.
*
* When the buffer is full, the flush_func is called. The flush_func
* can return -1 to indicate that no further output should be attempted,
* and apr_vformatter will return immediately with -1. Otherwise
* the flush_func should flush the buffer in whatever manner is
* appropriate, re apr_pool_t initialize curpos and endpos, and return 0.
*
* Note that flush_func is only invoked as a result of attempting to
* write another byte at curpos when curpos >= endpos. So for

```

```

* example, it's possible when the output exactly matches the buffer
* space available that curpos == endpos will be true when
* apr_vformatter returns.
*
* apr_vformatter does not call out to any other code, it is entirely
* self-contained. This allows the callers to do things which are
* otherwise "unsafe". For example, apr_psprintf uses the "scratch"
* space at the unallocated end of a block, and doesn't actually
* complete the allocation until apr_vformatter returns. apr_psprintf
* would be completely broken if apr_vformatter were to call anything
* that used this same pool. Similarly http_bprintf() uses the "scratch"
* space at the end of its output buffer, and doesn't actually note
* that the space is in use until it either has to flush the buffer
* or until apr_vformatter returns.
* </PRE>
*/
APR_DECLARE(int) apr_vformatter(int (*flush_func)(apr_vformatter_buff_t *b),
    apr_vformatter_buff_t *c, const char *fmt,
    va_list ap);

/**
 * Display a prompt and read in the password from stdin.
 * @param prompt The prompt to display
 * @param pwbuf Buffer to store the password
 * @param bufsize The length of the password buffer.
 * @remark If the password entered must be truncated to fit in
 * the provided buffer, APR_ENAMETOOLONG will be returned.
 * Note that the bufsize paramater is passed by reference for no
 * reason; its value will never be modified by the apr_password_get()
 * function.
 */
APR_DECLARE(apr_status_t) apr_password_get(const char *prompt, char *pwbuf,
    apr_size_t *bufsize);

/** @} */

/**
 * @defgroup apr_ctype ctype functions
 * These macros allow correct support of 8-bit characters on systems which
 * support 8-bit characters. Pretty dumb how the cast is required, but
 * that's legacy libc for ya. These new macros do not support EOF like
 * the standard macros do. Tough.
 * @{
 */
/** @see isalnum */
#define apr_isalnum(c) (isalnum(((unsigned char)(c))))
/** @see isalpha */
#define apr_isalpha(c) (isalpha(((unsigned char)(c))))
/** @see iscntrl */
#define apr_iscntrl(c) (iscntrl(((unsigned char)(c))))
/** @see isdigit */
#define apr_isdigit(c) (isdigit(((unsigned char)(c))))
/** @see isgraph */
#define apr_isgraph(c) (isgraph(((unsigned char)(c))))

```

```

/** @see islower*/
#define apr_islower(c) (islower(((unsigned char)(c))))
/** @see isascii */
#ifdef isascii
#define apr_isascii(c) (isascii(((unsigned char)(c))))
#else
#define apr_isascii(c) (((c) & ~0x7f)==0)
#endif
/** @see isprint */
#define apr_isprint(c) (isprint(((unsigned char)(c))))
/** @see ispunct */
#define apr_ispunct(c) (ispunct(((unsigned char)(c))))
/** @see isspace */
#define apr_isspace(c) (isspace(((unsigned char)(c))))
/** @see isupper */
#define apr_isupper(c) (isupper(((unsigned char)(c))))
/** @see isxdigit */
#define apr_isxdigit(c) (isxdigit(((unsigned char)(c))))
/** @see tolower */
#define apr_tolower(c) (tolower(((unsigned char)(c))))
/** @see toupper */
#define apr_toupper(c) (toupper(((unsigned char)(c))))

/** @} */

#ifdef __cplusplus
}
#endif

#endif/* ! APR_LIB_H */
        Apache License
        Version 2.0, January 2004
        http://www.apache.org/licenses/

```

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their

Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify

the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

APACHE PORTABLE RUNTIME SUBCOMPONENTS:

The Apache Portable Runtime includes a number of subcomponents with
separate copyright notices and license terms. Your use of the source
code for these subcomponents is subject to the terms and conditions
of the following licenses.

From strings/apr_fnmatch.c, include/apr_fnmatch.h, misc/unix/getopt.c,
file_io/unix/mktemp.c, strings/apr_strings.c:

```
/*
 * Copyright (c) 1987, 1993, 1994
 *   The Regents of the University of California. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 *    This product includes software developed by the University of
 *    California, Berkeley and its contributors.
 * 4. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
```


* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.

From network_io/unix/inet_ntop.c, network_io/unix/inet_pton.c:

/* Copyright (c) 1996 by Internet Software Consortium.

*

* Permission to use, copy, modify, and distribute this software for any
* purpose with or without fee is hereby granted, provided that the above
* copyright notice and this permission notice appear in all copies.

*

* THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
* ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
* CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
* DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
* PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
* ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
* SOFTWARE.

From dso/aix/dso.c:

* Based on libdl (dlfcn.c/dlfcn.h) which is
* Copyright (c) 1992,1993,1995,1996,1997,1988
* Jens-Uwe Mager, Helios Software GmbH, Hannover, Germany.

*

* Not derived from licensed software.

*

* Permission is granted to freely use, copy, modify, and redistribute
* this software, provided that the author is not construed to be liable
* for any results of using the software, alterations are clearly marked
* as such, and this notice is not modified.

From strings/apr_strnatcmp.c, include/apr_strings.h:

strnatcmp.c -- Perform 'natural order' comparisons of strings in C.
Copyright (C) 2000 by Martin Pool <mbp@humblebug.org.au>

This software is provided 'as-is', without any express or implied
warranty. In no event will the authors be held liable for any damages
arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
claim that you wrote the original software. If you use this software
in a product, an acknowledgment in the product documentation would be
appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be
misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

From strings/apr_sprintf.c:

```
*  
* cvt - IEEE floating point formatting routines.  
*   Derived from UNIX V7, Copyright(C) Caldera International Inc.  
*
```

Copyright(C) Caldera International Inc. 2001-2002. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code and documentation must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed or owned by Caldera International, Inc.

Neither the name of Caldera International, Inc. nor the names of other contributors may be used to endorse or promote products derived from this software without specific prior written permission.

USE OF THE SOFTWARE PROVIDED FOR UNDER THIS LICENSE BY CALDERA INTERNATIONAL, INC. AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL CALDERA INTERNATIONAL, INC. BE LIABLE FOR ANY DIRECT, INDIRECT INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.5 attr

1.5.1 Available under license

```

/*
 * Copyright (c) 2000-2002,2004 Silicon Graphics, Inc.
 * All Rights Reserved.
 *
 * This program is free software: you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */

#include <sys/types.h>
#include <sys/param.h>
#include <sys/stat.h>
#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <locale.h>

#include <attr/attributes.h>
#include "config.h"

#define SETOP1 /* do a SET operation */
#define GETOP2 /* do a GET operation */
#define REMOVEOP3 /* do a REMOVE operation */
#define LISTOP4 /* do a LIST operation */

#define BUFSIZE(60*1024) /* buffer size for LIST operations */

static char *progrname;

void
usage(void)
{
    fprintf(stderr, _("%s [-LRSq] -s attrname [-V attrvalue] pathname # set value\n"
    "    %s [-LRSq] -g attrname pathname          # get value\n"

```

```

"    %s [-LRSq] -r attrname pathname          # remove attr\n"
"    %s [-LRq] -l pathname                    # list attrs \n"
"    -s reads a value from stdin and -g writes a value to stdout\n"),
progname, progname, progname, progname);
exit(1);
}

int
main(int argc, char **argv)
{
char *attrname, *attrvalue, *filename, *buffer;
int attrlength, attrflags;
int opflag, i, ch, error, follow, verbose, rootflag, secureflag;
attrlist_t *alist;
attrlist_ent_t *aep;
attrlist_cursor_t cursor;

progname = basename(argv[0]);

setlocale(LC_CTYPE, "");
setlocale(LC_MESSAGES, "");
bindtextdomain(PACKAGE, LOCALEDIR);
textdomain(PACKAGE);

/*
 * Pick up and validate the arguments.
 */
verbose = 1;
follow = opflag = rootflag = secureflag = 0;
attrname = attrvalue = NULL;
while (((ch = getopt(argc, argv, "s:V:g:r:lqLRS")) != EOF) {
switch (ch) {
case 's':
if ((opflag != 0) && (opflag != SETOP)) {
fprintf(stderr,
_("Only one of -s, -g, -r, or -l allowed\n"));
usage();
}
opflag = SETOP;
attrname = optarg;
break;
case 'V':
if ((opflag != 0) && (opflag != SETOP)) {
fprintf(stderr, _("-V only allowed with -s\n"));
usage();
}
opflag = SETOP;
attrvalue = optarg;
break;
case 'g':
if (opflag) {
fprintf(stderr,
_("Only one of -s, -g, -r, or -l allowed\n"));
usage();
}

```

```

    opflag = GETOP;
    attrname = optarg;
    break;
    case 'r':
    if (opflag) {
    fprintf(stderr,
    _("Only one of -s, -g, -r, or -l allowed\n"));
    usage();
    }
    opflag = REMOVEOP;
    attrname = optarg;
    break;
    case 'l':
    if (opflag) {
    fprintf(stderr,
    _("Only one of -s, -g, -r, or -l allowed\n"));
    usage();
    }
    opflag = LISTOP;
    break;
    case 'L':
    follow++;
    break;
    case 'R':
    rootflag++;
    break;
    case 'S':
    secureflag++;
    break;
    case 'q':
    verbose = 0;
    break;
    default:
    fprintf(stderr, _("Unrecognized option: %c\n"),
    (char)ch);
    usage();
    break;
    }
    }
    if (optind != argc-1) {
    fprintf(stderr, _("A filename to operate on is required\n"));
    usage();
    }
    filename = argv[optind];

    attrflags = ((!follow ? ATTR_DONTFOLLOW : 0) |
    (secureflag ? ATTR_SECURE : 0) |
    (rootflag ? ATTR_ROOT : 0));
    /*
    * Break out into option-specific processing.
    */
    switch (opflag) {
    case SETOP:
    if (attrvalue == NULL) {

```

```

attrvalue = malloc(ATTR_MAX_VALUELEN);
if (attrvalue == NULL) {
    perror("malloc");
    exit(1);
}
attrlength =
fread(attrvalue, 1, ATTR_MAX_VALUELEN, stdin);
} else {
attrlength = strlen(attrvalue);
}
error = attr_set(filename, attrname, attrvalue,
    attrlength, attrflags);
if (error) {
    perror("attr_set");
    fprintf(stderr, _("Could not set \"%s\" for %s\n"),
        attrname, filename);
    exit(1);
}
if (verbose) {
    printf(_("Attribute \"%s\" set to a %d byte value "
        "for %s:\n"), attrname, attrlength, filename);
    fwrite(attrvalue, 1, attrlength, stdout);
    printf("\n");
}
break;

case GETOP:
attrvalue = malloc(ATTR_MAX_VALUELEN);
if (attrvalue == NULL) {
    perror("malloc");
    exit(1);
}
attrlength = ATTR_MAX_VALUELEN;
error = attr_get(filename, attrname, attrvalue,
    &attrlength, attrflags);
if (error) {
    perror("attr_get");
    fprintf(stderr, _("Could not get \"%s\" for %s\n"),
        attrname, filename);
    exit(1);
}
if (verbose) {
    printf(_("Attribute \"%s\" had a %d byte value "
        "for %s:\n"), attrname, attrlength, filename);
}
fwrite(attrvalue, 1, attrlength, stdout);
if (verbose) {
    printf("\n");
}
break;

case REMOVEOP:
error = attr_remove(filename, attrname, attrflags);
if (error) {
    perror("attr_remove");
}

```

```

fprintf(stderr, _("Could not remove \"%s\" for %s\n"),
attrname, filename);
exit(1);
}
break;

```

```

case LISTOP:
if ((buffer = malloc(BUFSIZE)) == NULL) {
perror("malloc");
exit(1);
}
bzero((char *)&cursor, sizeof(cursor));
do {
error = attr_list(filename, buffer, BUFSIZE,
attrflags, &cursor);
if (error) {
perror("attr_list");
fprintf(stderr,
_("Could not list \"%s\" for %s\n"),
attrname, filename);
exit(1);
}

```

```

alist = (attrlist_t *)buffer;
for (i = 0; i < alist->al_count; i++) {
aep = (attrlist_ent_t *)&buffer[ alist->al_offset[i] ];
if (verbose) {
printf(
_("Attribute \"%s\" has a %d byte value for %s\n"),
aep->a_name, aep->a_valuelen,
filename);
} else {
printf("%s\n", aep->a_name);
}
} while (alist->al_more);
break;

```

```

default:
fprintf(stderr,
_("At least one of -s, -g, -r, or -l is required\n"));
usage();
break;
}

```

```

return(0);
}

```

Most components of the "attr" package are licensed under Version 2.1 of the GNU Lesser General Public License (see COPYING.LGPL).

Some components (as annotated in the source) are licensed under Version 2 of the GNU General Public License (see below),

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under

these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component

itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the

integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES,

INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if

necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]
Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a

work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.
(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the

ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files

in the Library will not necessarily be able to recompile the application to use the modified definitions.)

- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy. For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the

section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the library's name and an idea of what it does.
Copyright (C) year name of author

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in
the library 'Frob' (a library for tweaking knobs) written
by James Random Hacker.

signature of Ty Coon, 1 April 1990
Ty Coon, President of Vice
That's all there is to it!

```
/*
 * Copyright (c) 2001-2003,2005 Silicon Graphics, Inc.
 * All Rights Reserved.
 *
 * This program is free software: you can redistribute it and/or modify it
 * under the terms of the GNU Lesser General Public License as published
 * by the Free Software Foundation, either version 2.1 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */
```

```
#include <errno.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>

#include <attr/xattr.h>
#include <attr/attributes.h>

#undef MAXNAMELEN
#define MAXNAMELEN 256
#undef MAXLISTLEN
#define MAXLISTLEN 65536

#undef roundup
#define roundup(x,y) (((x)+((y)-1))/(y))*(y)

static const char *user_name = "user.";
static const char *secure_name = "security.";
static const char *trusted_name = "trusted.";
static const char *xfsroot_name = "xfsroot.";

/*
 * Convert IRIX API components into Linux/XFS API components,
 * and vice-versa.
 */
static int
api_convert(char *name, const char *irixname, int irixflags, int compat)
{
    if (strlen(irixname) >= MAXNAMELEN) {
        errno = EINVAL;
        return -1;
    }
    if (irixflags & ATTR_ROOT) {
        if (compat)
            strcpy(name, xfsroot_name);
        else
            strcpy(name, trusted_name);
    } else if (irixflags & ATTR_SECURE) {
        strcpy(name, secure_name);
    } else {
        strcpy(name, user_name);
    }
    strcat(name, irixname);
    return 0;
}

static int
api_unconvert(char *name, const char *linuxname, int irixflags)
{
    int type, length;

    length = strlen(user_name);
```

```

    if (strncmp(linuxname, user_name, length) == 0) {
        type = 0; /*ATTR_USER*/
        goto found;
    }
    length = strlen(secure_name);
    if (strncmp(linuxname, secure_name, length) == 0) {
        type = ATTR_SECURE;
        goto found;
    }
    length = strlen(trusted_name);
    if (strncmp(linuxname, trusted_name, length) == 0) {
        type = ATTR_ROOT;
        goto found;
    }
    length = strlen(xfsroot_name);
    if (strncmp(linuxname, xfsroot_name, length) == 0) {
        type = ATTR_ROOT;
        goto found;
    }
    return 1;

found:
    if ((irixflags & ATTR_SECURE) != 0 && (type != ATTR_SECURE))
        return 1;
    if ((irixflags & ATTR_ROOT) != 0 && (type != ATTR_ROOT))
        return 1;
    strcpy(name, linuxname + length);
    return 0;
}

int
attr_get(const char *path, const char *attrname, char *attrvalue,
         int *valuelength, int flags)
{
    int c, compat;
    char name[MAXNAMELEN+16];

    for (compat = 0; compat < 2; compat++) {
        if ((c = api_convert(name, attrname, flags, compat)) < 0)
            return c;
        if (flags & ATTR_DONTFOLLOW)
            c = lgetxattr(path, name, attrvalue, *valuelength);
        else
            c = getxattr(path, name, attrvalue, *valuelength);
        if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        break;
    }
    if (c < 0)
        return c;
    *valuelength = c;
    return 0;
}

```

```
int
attr_getf(int fd, const char *attrname, char *attrvalue,
    int *valuelength, int flags)
{
    int c, compat;
    char name[MAXNAMELEN+16];

    for (compat = 0; compat < 2; compat++) {
        if ((c = api_convert(name, attrname, flags, compat)) < 0)
            return c;
        c = fgetxattr(fd, name, attrvalue, *valuelength);
        if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        break;
    }
    if (c < 0)
        return c;
    *valuelength = c;
    return 0;
}

int
attr_set(const char *path, const char *attrname, const char *attrvalue,
    const int valuelength, int flags)
{
    int c, compat, lflags = 0;
    char name[MAXNAMELEN+16];
    void *buffer = (void *)attrvalue;

    if (flags & ATTR_CREATE)
        lflags = XATTR_CREATE;
    else if (flags & ATTR_REPLACE)
        lflags = XATTR_REPLACE;

    for (compat = 0; compat < 2; compat++) {
        if ((c = api_convert(name, attrname, flags, compat)) < 0)
            return c;
        if (flags & ATTR_DONTFOLLOW)
            c = lsetxattr(path, name, buffer, valuelength, lflags);
        else
            c = setxattr(path, name, buffer, valuelength, lflags);
        if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        break;
    }
    return c;
}

int
attr_setf(int fd, const char *attrname,
    const char *attrvalue, const int valuelength, int flags)
{
    int c, compat, lflags = 0;
    char name[MAXNAMELEN+16];
```

```

void *buffer = (void *)attrvalue;

if (flags & ATTR_CREATE)
    lflags = XATTR_CREATE;
else if (flags & ATTR_REPLACE)
    lflags = XATTR_REPLACE;

for (compat = 0; compat < 2; compat++) {
    if ((c = api_convert(name, attrname, flags, compat)) < 0)
        return c;
    c = fsetxattr(fd, name, buffer, valuelength, lflags);
    if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
        continue;
    break;
}
return c;
}

int
attr_remove(const char *path, const char *attrname, int flags)
{
    int c, compat;
    char name[MAXNAMELEN+16];

    for (compat = 0; compat < 2; compat++) {
        if ((c = api_convert(name, attrname, flags, compat)) < 0)
            return c;
        if (flags & ATTR_DONTFOLLOW)
            c = lremovexattr(path, name);
        else
            c = removexattr(path, name);
        if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        break;
    }
    return c;
}

int
attr_removef(int fd, const char *attrname, int flags)
{
    int c, compat;
    char name[MAXNAMELEN+16];

    for (compat = 0; compat < 2; compat++) {
        if ((c = api_convert(name, attrname, flags, compat)) < 0)
            return c;
        c = fremovexattr(fd, name);
        if (c < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        break;
    }
    return c;
}

```



```

/*
 * Helper routine for attr_list functions.
 */

static int
attr_list_pack(const char *name, const int valuelen,
char *buffer, const int buffersize,
int *start_offset, int *end_offset)
{
    attrlist_ent_t *aentp;
    attrlist_t *alist = (attrlist_t *)buffer;
    int size = roundup(strlen(name) + 1 + sizeof(aentp->a_valuelen), 8);

    if ((*end_offset - size) < (*start_offset + sizeof(alist->al_count))) {
        alist->al_more = 1;
        return 1;
    }

    *end_offset -= size;
    aentp = (attrlist_ent_t *)&buffer[*end_offset];
    aentp->a_valuelen = valuelen;
    strncpy(aentp->a_name, name, size - sizeof(aentp->a_valuelen));

    *start_offset += sizeof(alist->al_offset);
    alist->al_offset[alist->al_count] = *end_offset;
    alist->al_count++;
    return 0;
}

int
attr_list(const char *path, char *buffer, const int buffersize, int flags,
    attrlist_cursor_t *cursor)
{
    const char *l;
    int length, vlength, count = 0;
    char lbuf[MAXLISTLEN];
    char name[MAXNAMELEN+16];
    int start_offset, end_offset;

    if (buffersize < sizeof(attrlist_t)) {
        errno = EINVAL;
        return -1;
    }
    bzero(buffer, sizeof(attrlist_t));

    if (flags & ATTR_DONTFOLLOW)
        length = llistxattr(path, lbuf, sizeof(lbuf));
    else
        length = listxattr(path, lbuf, sizeof(lbuf));
    if (length <= 0)
        return length;

    start_offset = sizeof(attrlist_t);

```

```

end_offset = buffersize & ~(8-1);/* 8 byte align */

for (l = lbuf; l != lbuf + length; l = strchr(l, '\0') + 1) {
    if (api_unconvert(name, l, flags))
        continue;
    if (flags & ATTR_DONTFOLLOW)
        vlength = lgetxattr(path, l, NULL, 0);
    else
        vlength = getxattr(path, l, NULL, 0);
    if (vlength < 0 && (errno == ENOATTR || errno == ENOTSUP))
        continue;
    if (count++ < cursor->opaque[0])
        continue;
    if (attr_list_pack(name, vlength, buffer, buffersize,
        &start_offset, &end_offset)) {
        cursor->opaque[0] = count;
        break;
    }
}
return 0;
}

int
attr_listf(int fd, char *buffer, const int buffersize, int flags,
    attrlist_cursor_t *cursor)
{
    const char *l;
    int length, vlength, count = 0;
    char lbuf[MAXLISTLEN];
    char name[MAXNAMELEN+16];
    int start_offset, end_offset;

    if (buffersize < sizeof(attrlist_t)) {
        errno = EINVAL;
        return -1;
    }
    bzero(buffer, sizeof(attrlist_t));

    length = flistxattr(fd, lbuf, sizeof(lbuf));
    if (length < 0)
        return length;

    start_offset = sizeof(attrlist_t);
    end_offset = buffersize & ~(8-1);/* 8 byte align */

    for (l = lbuf; l != lbuf + length; l = strchr(l, '\0') + 1) {
        if (api_unconvert(name, l, flags))
            continue;
        vlength = fgetxattr(fd, l, NULL, 0);
        if (vlength < 0 && (errno == ENOATTR || errno == ENOTSUP))
            continue;
        if (count++ < cursor->opaque[0])
            continue;
        if (attr_list_pack(name, vlength, buffer, buffersize,
            &start_offset, &end_offset)) {

```

```

    cursor->opaque[0] = count;
    break;
}
}
return 0;
}

/*
 * Helper routines for the attr_multi functions. In IRIX, the
 * multi routines are a single syscall - in Linux, we break em
 * apart in userspace and make individual syscalls for each.
 */

static int
attr_single(const char *path, attr_multiop_t *op, int flags)
{
    int r = -1;

    errno = -EINVAL;
    flags |= op->am_flags;
    if (op->am_opcode & ATTR_OP_GET)
        r = attr_get(path, op->am_attrname, op->am_attrvalue,
            &op->am_length, flags);
    else if (op->am_opcode & ATTR_OP_SET)
        r = attr_set(path, op->am_attrname, op->am_attrvalue,
            op->am_length, flags);
    else if (op->am_opcode & ATTR_OP_REMOVE)
        r = attr_remove(path, op->am_attrname, flags);
    return r;
}

static int
attr_singlef(const int fd, attr_multiop_t *op, int flags)
{
    int r = -1;

    errno = -EINVAL;
    flags |= op->am_flags;
    if (op->am_opcode & ATTR_OP_GET)
        r = attr_getf(fd, op->am_attrname, op->am_attrvalue,
            &op->am_length, flags);
    else if (op->am_opcode & ATTR_OP_SET)
        r = attr_setf(fd, op->am_attrname, op->am_attrvalue,
            op->am_length, flags);
    else if (op->am_opcode & ATTR_OP_REMOVE)
        r = attr_removef(fd, op->am_attrname, flags);
    return r;
}

/*
 * Operate on multiple attributes of the same object simultaneously
 *
 * From the manpage: "attr_multi will fail if ... a bit other than

```

```

    * ATTR_DONTFOLLOW was set in the flag argument." flags must be
    * checked here as they are not passed into the kernel.
    */
int
attr_multi(const char *path, attr_multiop_t *multiops, int count, int flags)
{
    int i, tmp, r = -1;

    errno = EINVAL;
    if ((flags & ATTR_DONTFOLLOW) != flags)
        return r;

    r = errno = 0;
    for (i = 0; i < count; i++) {
        tmp = attr_single(path, &multiops[i], flags);
        if (tmp) r = tmp;
    }
    return r;
}

int
attr_multif(int fd, attr_multiop_t *multiops, int count, int flags)
{
    int i, tmp, r = -1;

    errno = EINVAL;
    if ((flags & ATTR_DONTFOLLOW) != flags)
        return r;

    r = errno = 0;
    for (i = 0; i < count; i++) {
        tmp = attr_singlef(fd, &multiops[i], flags);
        if (tmp) r = tmp;
    }
    return r;
}

```

1.6 base-files

1.6.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.7 base-passwd

1.7.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.8 bash

1.8.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.9 beecrypt

1.9.1 Available under license

```

/*
 * Copyright (c) 1999, 2000, 2001, 2002 X-Way Rights BV
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 */

/*!\file beecrypt.h
 * \brief BeeCrypt API, headers.
 *
 * These API functions provide an abstract way for using most of
 * the various algorithms implemented by the library.
 *
 * \author Bob Deblier <bob.deblier@telenet.be>
 * \ingroup ES_m PRNG_m HASH_m HMAC_m BC_m
 */

#ifndef _BEECRYPT_H
#define _BEECRYPT_H

#include "beecrypt/api.h"

#include "beecrypt/memchunk.h"
#include "beecrypt/mpnumber.h"

/*
 * Entropy Sources
 */

/*!\typedef entropyNext
 * \brief Prototype definition for an entropy-generating function.
 * \ingroup ES_m
 */
typedef int (*entropyNext)(byte*, size_t);

/*!\brief This struct holds information and pointers to code specific to each

```



```

    * source of entropy.
    * \ingroup ES_m
    */
#ifdef __cplusplus
struct BEECRYPTAPI entropySource
#else
struct _entropySource
#endif
{
    /*!\var name
    * \brief The entropy source's name.
    */
    const char*name;
    /*!\var next
    * \brief Points to the function which produces the entropy.
    */
    const entropyNextnext;
};

#ifdef __cplusplus
typedef struct _entropySource entropySource;
#endif

#ifdef __cplusplus
extern "C" {
#endif

/*!\fn int entropySourceCount()
* \brief This function returns the number of entropy sources implemented by
* the library.
* \return The number of implemented entropy sources.
*/
BEECRYPTAPI
intentropySourceCount(void);

/*!\fn const entropySource* entropySourceGet(int n)
* \brief This function returns the \a n -th entropy source implemented by
* the library.
* \param n Index of the requested entropy source; legal values are 0
* through entropySourceCount() - 1.
* \return A pointer to an entropy source or null, if the index was out of
* range.
*/
BEECRYPTAPI
const entropySource*entropySourceGet(int n);

/*!\fn const entropySource* entropySourceFind(const char* name)
* \brief This function returns the entropy source specified by the given name.
* \param name Name of the requested entropy source.
* \return A pointer to an entropy source or null, if the name wasn't found.
*/
BEECRYPTAPI
const entropySource*entropySourceFind(const char* name);

```

```

/*!\fn const entropySource* entropySourceDefault()
 * \brief This functions returns the default entropy source; the default value
 * can be specified by setting environment variable BEECRYPT_ENTROPY.
 * \return A pointer to an entropy source or null, in case an error occurred.
 */
BEECRYPTAPI
const entropySource*entropySourceDefault(void);

/*!\fn int entropyGatherNext(byte* data, size_t size)
 * \brief This function gathers \a size bytes of entropy into \a data.
 *
 * Unless environment variable BEECRYPT_ENTROPY is set, this function will
 * try each successive entropy source to gather up the requested amount.
 *
 * \param data Points to where the entropy should be stored.
 * \param size Indicates how many bytes of entropy should be gathered.
 * \retval 0 On success.
 * \retval -1 On failure.
 */
BEECRYPTAPI
intentropyGatherNext(byte*, size_t);

#ifdef __cplusplus
}
#endif

/*
 * Pseudo-random Number Generators
 */

typedef void randomGeneratorParam;

typedef int (*randomGeneratorSetup )(randomGeneratorParam*);
typedef int (*randomGeneratorSeed )(randomGeneratorParam*, const byte*, size_t);
typedef int (*randomGeneratorNext )(randomGeneratorParam*, byte*, size_t);
typedef int (*randomGeneratorCleanup)(randomGeneratorParam*);

/*
 * The struct 'randomGenerator' holds information and pointers to code specific
 * to each random generator. Each specific random generator MUST be written to
 * be multithread safe.
 *
 * WARNING: each randomGenerator, when used in cryptographic applications, MUST
 * be guaranteed to be of suitable quality and strength (i.e. don't use the
 * random() function found in most UN*X-es).
 *
 * Multiple instances of each randomGenerator can be used (even concurrently),
 * provided they each use their own randomGeneratorParam parameters, a chunk
 * of memory which must be at least as large as indicated by the paramsize
 * field.
 */

/*!\brief This struct holds information and pointers to code specific to each
 * pseudo-random number generator.

```

```

    * \ingroup PRNG_m
    */
#ifdef __cplusplus
struct BEECRYPTAPI randomGenerator
#else
struct _randomGenerator
#endif
{
    /*!\var name
    * \brief The random generator's name.
    */
    const char*name;
    /*!\var paramsize
    * \brief The size of the random generator's parameters.
    * \note The implementor should set this by using sizeof(<struct holding
    * random generator's parameters>).
    */
    const size_tparamsize;
    /*!\var setup
    * \brief Points to the setup function.
    */
    const randomGeneratorSetupsetup;
    /*!\var seed
    * \brief Points to the seeding function.
    */
    const randomGeneratorSeedseed;
    /*!\var seed
    * \brief Points to the function which generates the random data.
    */
    const randomGeneratorNextnext;
    /*!\var seed
    * \brief Points to the cleanup function.
    */
    const randomGeneratorCleanupcleanup;
};

#ifdef __cplusplus
typedef struct _randomGenerator randomGenerator;
#endif

/*
 * You can use the following functions to find random generators implemented by
 * the library:
 *
 * randomGeneratorCount returns the number of generators available.
 *
 * randomGeneratorGet returns the random generator with a given index (starting
 * at zero, up to randomGeneratorCount() - 1), or NULL if the index was out of
 * bounds.
 *
 * randomGeneratorFind returns the random generator with the given name, or
 * NULL if no random generator exists with that name.
 */

```

```

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
intrandomGeneratorCount(void);
BEECRYPTAPI
const randomGenerator*randomGeneratorGet(int);
BEECRYPTAPI
const randomGenerator*randomGeneratorFind(const char*);
BEECRYPTAPI
const randomGenerator*randomGeneratorDefault(void);

#ifdef __cplusplus
}
#endif

/*
 * The struct 'randomGeneratorContext' is used to contain both the functional
 * part (the randomGenerator), and its parameters.
 */

#ifdef __cplusplus
struct BEECRYPTAPI randomGeneratorContext
#else
struct _randomGeneratorContext
#endif
{
    const randomGenerator* rng;
    randomGeneratorParam* param;

#ifdef __cplusplus
    randomGeneratorContext();
    randomGeneratorContext(const randomGenerator*);
    ~randomGeneratorContext();
#endif
};

#ifdef __cplusplus
typedef struct _randomGeneratorContext randomGeneratorContext;
#endif

/*
 * The following functions can be used to initialize and free a
 * randomGeneratorContext. Initializing will allocate a buffer of the size
 * required by the randomGenerator, freeing will deallocate that buffer.
 */

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
int randomGeneratorContextInit(randomGeneratorContext*, const randomGenerator*);
BEECRYPTAPI

```

```

int randomGeneratorContextFree(randomGeneratorContext*);
BEECRYPTAPI
int randomGeneratorContextNext(randomGeneratorContext*, byte*, size_t);
BEECRYPTAPI
int randomGeneratorContextSeed(randomGeneratorContext*, const byte*, size_t);

#ifdef __cplusplus
}
#endif

/*
 * Hash Functions
 */

/*!typedef void hashFunctionParam
 * \ingroup HASH_m
 */
typedef void hashFunctionParam;

typedef int (*hashFunctionReset)(hashFunctionParam*);
typedef int (*hashFunctionUpdate)(hashFunctionParam*, const byte*, size_t);
typedef int (*hashFunctionDigest)(hashFunctionParam*, byte*);

/*
 * The struct 'hashFunction' holds information and pointers to code specific
 * to each hash function. Specific hash functions MAY be written to be
 * multithread-safe.
 *
 * NOTE: data MUST have a size (in bytes) of at least 'digestsize' as described
 * in the hashFunction struct.
 * NOTE: for safety reasons, after calling digest, each specific implementation
 * MUST reset itself so that previous values in the parameters are erased.
 */
#ifdef __cplusplus
struct BEECRYPTAPI hashFunction
#else
struct _hashFunction
#endif
{
    const char*name;
    const size_tparamsize; /* in bytes */
    const size_tblocksize; /* in bytes */
    const size_tdigestsize; /* in bytes */
    const hashFunctionResetreset;
    const hashFunctionUpdateupdate;
    const hashFunctionDigestdigest;
};

#ifdef __cplusplus
typedef struct _hashFunction hashFunction;
#endif

/*
 * You can use the following functions to find hash functions implemented by

```

```

* the library:
*
* hashFunctionCount returns the number of hash functions available.
*
* hashFunctionGet returns the hash function with a given index (starting
* at zero, up to hashFunctionCount() - 1), or NULL if the index was out of
* bounds.
*
* hashFunctionFind returns the hash function with the given name, or
* NULL if no hash function exists with that name.
*/

```

```

#ifdef __cplusplus
extern "C" {
#endif

```

```

BEECRYPTAPI
inthashFunctionCount(void);
BEECRYPTAPI
const hashFunction*hashFunctionGet(int);
BEECRYPTAPI
const hashFunction*hashFunctionFind(const char*);
BEECRYPTAPI
const hashFunction*hashFunctionDefault(void);

```

```

#ifdef __cplusplus
}
#endif

```

```

/*
* The struct 'hashFunctionContext' is used to contain both the functional
* part (the hashFunction), and its parameters.
*/

```

```

#ifdef __cplusplus
struct BEECRYPTAPI hashFunctionContext
#else
struct _hashFunctionContext
#endif
{
    const hashFunction* algo;
    hashFunctionParam* param;

```

```

#ifdef __cplusplus
hashFunctionContext();
hashFunctionContext(const hashFunction*);
~hashFunctionContext();
#endif
};

```

```

#ifndef __cplusplus
typedef struct _hashFunctionContext hashFunctionContext;
#endif

```

```

/*
* The following functions can be used to initialize and free a

```

```

    * hashFunctionContext. Initializing will allocate a buffer of the size
    * required by the hashFunction, freeing will deallocate that buffer.
    */

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
int hashFunctionContextInit(hashFunctionContext*, const hashFunction*);
BEECRYPTAPI
int hashFunctionContextFree(hashFunctionContext*);
BEECRYPTAPI
int hashFunctionContextReset(hashFunctionContext*);
BEECRYPTAPI
int hashFunctionContextUpdate(hashFunctionContext*, const byte*, size_t);
BEECRYPTAPI
int hashFunctionContextUpdateMC(hashFunctionContext*, const memchunk*);
BEECRYPTAPI
int hashFunctionContextUpdateMP(hashFunctionContext*, const mpnumber*);
BEECRYPTAPI
int hashFunctionContextDigest(hashFunctionContext*, byte*);
BEECRYPTAPI
int hashFunctionContextDigestMP(hashFunctionContext*, mpnumber*);
BEECRYPTAPI
int hashFunctionContextDigestMatch(hashFunctionContext*, const mpnumber*);

#ifdef __cplusplus
}
#endif

/*
 * Keyed Hash Functions, a.k.a. Message Authentication Codes
 */

/*!\typedef void keyedHashFunctionParam
 * \ingroup HMAC_m
 */
typedef void keyedHashFunctionParam;

typedef int (*keyedHashFunctionSetup )(keyedHashFunctionParam*, const byte*, size_t);
typedef int (*keyedHashFunctionReset )(keyedHashFunctionParam*);
typedef int (*keyedHashFunctionUpdate )(keyedHashFunctionParam*, const byte*, size_t);
typedef int (*keyedHashFunctionDigest )(keyedHashFunctionParam*, byte*);

/*
 * The struct 'keyedHashFunction' holds information and pointers to code
 * specific to each keyed hash function. Specific keyed hash functions MAY be
 * written to be multithread-safe.
 *
 * The struct field 'keybitsmin' contains the minimum number of bits a key
 * must contain, 'keybitsmax' the maximum number of bits a key may contain,
 * 'keybitsinc', the increment in bits that may be used between min and max.
 */

```

```

* NOTE: data must be at least have a bytesize of 'digestsize' as described
* in the keyedHashFunction struct.
* NOTE: for safety reasons, after calling digest, each specific implementation
* MUST reset itself so that previous values in the parameters are erased.
*/
#ifdef __cplusplus
struct BEECRYPTAPI keyedHashFunction
#else
struct _keyedHashFunction
#endif
{
    const char*name;
    const size_tparamsize;/* in bytes */
    const size_tblocksize;/* in bytes */
    const size_tdigestsize;/* in bytes */
    const size_tkeybitsmin;/* in bits */
    const size_tkeybitsmax;/* in bits */
    const size_tkeybitsinc;/* in bits */
    const keyedHashFunctionSetupsetup;
    const keyedHashFunctionResetreset;
    const keyedHashFunctionUpdateupdate;
    const keyedHashFunctionDigestdigest;
};

#ifdef __cplusplus
typedef struct _keyedHashFunction keyedHashFunction;
#endif

/*
* You can use the following functions to find keyed hash functions implemented
* by the library:
*
* keyedHashFunctionCount returns the number of keyed hash functions available.
*
* keyedHashFunctionGet returns the keyed hash function with a given index
* (starting at zero, up to keyedHashFunctionCount() - 1), or NULL if the index
* was out of bounds.
*
* keyedHashFunctionFind returns the keyed hash function with the given name,
* or NULL if no keyed hash function exists with that name.
*/

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
intkeyedHashFunctionCount(void);
BEECRYPTAPI
const keyedHashFunction*keyedHashFunctionGet(int);
BEECRYPTAPI
const keyedHashFunction*keyedHashFunctionFind(const char*);
BEECRYPTAPI
const keyedHashFunction*keyedHashFunctionDefault(void);

```



```

#ifdef __cplusplus
}
#endif

/*
 * The struct 'keyedHashFunctionContext' is used to contain both the functional
 * part (the keyedHashFunction), and its parameters.
 */
#ifdef __cplusplus
struct BEECRYPTAPI keyedHashFunctionContext
#else
struct _keyedHashFunctionContext
#endif
{
    const keyedHashFunction* algo;
    keyedHashFunctionParam* param;

#ifdef __cplusplus
    keyedHashFunctionContext();
    keyedHashFunctionContext(const keyedHashFunction*);
    ~keyedHashFunctionContext();
#endif
};

#ifndef __cplusplus
typedef struct _keyedHashFunctionContext keyedHashFunctionContext;
#endif

/*
 * The following functions can be used to initialize and free a
 * keyedHashFunctionContext. Initializing will allocate a buffer of the size
 * required by the keyedHashFunction, freeing will deallocate that buffer.
 */

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
int keyedHashFunctionContextInit(keyedHashFunctionContext*, const keyedHashFunction*);
BEECRYPTAPI
int keyedHashFunctionContextFree(keyedHashFunctionContext*);
BEECRYPTAPI
int keyedHashFunctionContextSetup(keyedHashFunctionContext*, const byte*, size_t);
BEECRYPTAPI
int keyedHashFunctionContextReset(keyedHashFunctionContext*);
BEECRYPTAPI
int keyedHashFunctionContextUpdate(keyedHashFunctionContext*, const byte*, size_t);
BEECRYPTAPI
int keyedHashFunctionContextUpdateMC(keyedHashFunctionContext*, const memchunk*);
BEECRYPTAPI
int keyedHashFunctionContextUpdateMP(keyedHashFunctionContext*, const mpnumber*);
BEECRYPTAPI
int keyedHashFunctionContextDigest(keyedHashFunctionContext*, byte*);

```

```

BEECRYPTAPI
int keyedHashFunctionContextDigestMP(keyedHashFunctionContext*, mpnumber*);
BEECRYPTAPI
int keyedHashFunctionContextDigestMatch(keyedHashFunctionContext*, const mpnumber*);

#ifdef __cplusplus
}
#endif

/*
 * Block ciphers
 */

/*!\enum cipherOperation
 * \brief Specifies whether to perform encryption or decryption.
 * \ingroup BC_m
 */
typedef enum
{
    NOCRYPT,
    ENCRYPT,
    DECRYPT
} cipherOperation;

/*!\typedef void blockCipherParam
 * \brief Placeholder type definition for blockcipher parameters.
 * \sa aesParam, blowfishParam.
 * \ingroup BC_m
 */
typedef void blockCipherParam;

/*!\brief Prototype definition for a setup function.
 * \ingroup BC_m
 */
typedef int (*blockCipherSetup )(blockCipherParam*, const byte*, size_t, cipherOperation);

/*!\typedef int (*blockCipherSetIV)(blockCipherParam* bp, const byte* iv)
 * \brief Prototype definition for an initialization vector setup function.
 * \param bp The blockcipher's parameters.
 * \param iv The blockciphers' IV value.
 * \note iv length must be equal to the cipher's block size.
 * \retval 0 on success.
 * \retval -1 on failure.
 * \ingroup BC_m
 */
typedef int (*blockCipherSetIV )(blockCipherParam*, const byte*);

/*!\typedef int (*blockCipherSetCTR)(blockCipherParam* bp, const byte* nivz, size_t counter)
 * \brief Prototype definition for an initialization vector setup function.
 * \param bp The blockcipher's parameters.
 * \param nivz The concatenation of the Nonce, IV and padding Zero bytes.
 * \param counter The blockciphers' counter value.
 * \note nivz length must be equal to the cipher's block size.
 * \retval 0 on success.
 * \retval -1 on failure.

```

```

* \ingroup BC_m
*/
typedef int (*blockCipherSetCTR)(blockCipherParam*, const byte*, size_t);

/*!\typedef int (*blockCipherRawcrypt)(blockCipherParam* bp, uint32_t* dst, const uint32_t* src)
* \brief Prototype for a \e raw encryption or decryption function.
* \param bp The blockcipher's parameters.
* \param dst The ciphertext address; must be aligned on 32-bit boundary.
* \param src The cleartext address; must be aligned on 32-bit boundary.
* \retval 0 on success.
* \retval -1 on failure.
* \ingroup BC_m
*/
typedef int (*blockCipherRawcrypt)(blockCipherParam*, uint32_t*, const uint32_t*);

/*!\typedef int (*blockCipherModcrypt)(blockCipherParam* bp, uint32_t* dst, const uint32_t* src, unsigned int
nblocks)
* \brief Prototype for a \e encryption or decryption function which operates
*      on multiple blocks in a certain mode.
* \param bp The blockcipher's parameters.
* \param dst The ciphertext address; must be aligned on 32-bit boundary.
* \param src The cleartext address; must be aligned on 32-bit boundary.
* \param nblocks The number of blocks to process.
* \retval 0 on success.
* \retval -1 on failure.
* \ingroup BC_m
*/
typedef int (*blockCipherModcrypt)(blockCipherParam*, uint32_t*, const uint32_t*, unsigned int);

typedef uint32_t* (*blockCipherFeedback)(blockCipherParam*);

typedef struct
{
    const blockCipherRawcrypt encrypt;
    const blockCipherRawcrypt decrypt;
} blockCipherRaw;

typedef struct
{
    const blockCipherModcrypt encrypt;
    const blockCipherModcrypt decrypt;
} blockCipherMode;

/*!\brief Holds information and pointers to code specific to each cipher.
*
* Specific block ciphers \e may be written to be multithread-safe.
*
* \ingroup BC_m
*/
#ifdef __cplusplus
struct BEECRYPTAPI blockCipher
#else
struct _blockCipher

```

```

#endif
{
/*!\var name
 * \brief The blockcipher's name.
 */
const char*name;
/*!\var paramsize
 * \brief The size of the parameters required by this cipher, in bytes.
 */
const size_tparamsize;
/*!\var blocksize
 * \brief The size of one block of data, in bytes.
 */
const size_tblocksize;
/*!\var keybitsmin
 * \brief The minimum number of key bits.
 */
const size_tkeybitsmin;
/*!\var keybitsmax
 * \brief The maximum number of key bits.
 */
const size_tkeybitsmax;
/*!\var keybitsinc
 * \brief The allowed increment in key bits between min and max.
 * \see keybitsmin and keybitsmax.
 */
const size_tkeybitsinc;
/*!\var setup
 * \brief Pointer to the cipher's setup function.
 */
const blockCipherSetupsetup;
/*!\var setiv
 * \brief Pointer to the cipher's initialization vector setup function.
 */
const blockCipherSetIVsetiv;
/*!\var setctr
 * \brief Pointer to the cipher's ctr setup function.
 */
const blockCipherSetCTRsetctr;
/*!\var getfb
 * \brief Pointer to the cipher's feedback-returning function.
 */
const blockCipherFeedbackgetfb;
/*!\var raw
 * \brief The cipher's raw functions.
 */
const blockCipherRawraw;
/*!\var ecb
 * \brief The cipher's ECB functions.
 */
const blockCipherModeecb;
/*!\var cbc
 * \brief The cipher's CBC functions.
 */
const blockCipherModecbc;

```

```

/*!\var ctr
 * \brief The cipher's CTR functions.
 */
const blockCipherModectr;
};

#ifdef __cplusplus
typedef struct _blockCipher blockCipher;
#endif

#ifdef __cplusplus
extern "C" {
#endif

/*!\fn int blockCipherCount()
 * \brief This function returns the number of blockciphers implemented
 * by the library.
 * \return The number of implemented blockciphers.
 */
BEECRYPTAPI
int blockCipherCount(void);

/*!\fn const blockCipher* blockCipherGet(int n)
 * \brief This function returns the \a n -th blockcipher implemented by
 * the library.
 * \param n Index of the requested blockcipher; legal values are 0
 * through blockCipherCount() - 1.
 * \return A pointer to a blockcipher or null, if the index was out of
 * range.
 */
BEECRYPTAPI
const blockCipher* blockCipherGet(int);

/*!\fn const blockCipher* blockCipherFind(const char* name)
 * \brief This function returns the blockcipher specified by the given name.
 * \param name Name of the requested blockcipher.
 * \return A pointer to a blockcipher or null, if the name wasn't found.
 */
BEECRYPTAPI
const blockCipher* blockCipherFind(const char*);

/*!\fn const blockCipher* blockCipherDefault()
 * \brief This functions returns the default blockcipher; the default value
 * can be specified by setting environment variable BEECRYPT_CIPHER.
 * \return A pointer to a blockcipher or null, in case an error occurred.
 */
BEECRYPTAPI
const blockCipher* blockCipherDefault(void);

#ifdef __cplusplus
}
#endif

/*!\brief Holds a pointer to a blockcipher as well as its parameters.

```

```

    * \warning A context can be used by only one thread at the same time.
    * \ingroup BC_m
    */
#ifdef __cplusplus
struct BEECRYPTAPI blockCipherContext
#else
struct _blockCipherContext
#endif
{
    /*!\var algo
    * \brief Pointer to a blockCipher.
    */
    const blockCipher*algo;
    /*!\var param
    * \brief Pointer to the parameters used by algo.
    */
    blockCipherParam*param;
    /*!\var op
    */
    cipherOperationop;

#ifdef __cplusplus
    blockCipherContext();
    blockCipherContext(const blockCipher*);
    ~blockCipherContext();
#endif
};

#ifdef __cplusplus
typedef struct _blockCipherContext blockCipherContext;
#endif

/*
 * The following functions can be used to initialize and free a
 * blockCipherContext. Initializing will allocate a buffer of the size
 * required by the blockCipher, freeing will deallocate that buffer.
 */

#ifdef __cplusplus
extern "C" {
#endif

BEECRYPTAPI
int blockCipherContextInit(blockCipherContext*, const blockCipher*);

BEECRYPTAPI
int blockCipherContextSetup(blockCipherContext*, const byte*, size_t, cipherOperation);

BEECRYPTAPI
int blockCipherContextSetIV(blockCipherContext*, const byte*);

BEECRYPTAPI
int blockCipherContextSetCTR(blockCipherContext*, const byte*, size_t);

BEECRYPTAPI

```

```
int blockCipherContextFree(blockCipherContext*);

BEECRYPTAPI
int blockCipherContextECB(blockCipherContext*, uint32_t*, const uint32_t*, int);

BEECRYPTAPI
int blockCipherContextCBC(blockCipherContext*, uint32_t*, const uint32_t*, int);

BEECRYPTAPI
int blockCipherContextCTR(blockCipherContext*, uint32_t*, const uint32_t*, int);

BEECRYPTAPI
int blockCipherContextValidKeylen(blockCipherContext*, size_t);

#ifdef __cplusplus
}
#endif

#endif
```

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the

source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion

of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete

machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes

make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Hereny it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Section

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it

does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If

identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a

copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies,

or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries,

so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest

possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.10 bind

1.10.1 Available under license

Copyright (C) 2004-2015 Internet Systems Consortium, Inc. ("ISC")
Copyright (C) 1996-2003 Internet Software Consortium.

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Portions of this code release fall under one or more of the following Copyright notices. Please see individual source files for details.

For binary releases also see: OpenSSL-LICENSE.

Copyright (C) 1996-2001 Nominum, Inc.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND NOMINUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL NOMINUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (C) 1995-2000 by Network Associates, Inc.

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND ISC AND NETWORK ASSOCIATES DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES

WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (C) 2002 Stichting NLnet, Netherlands, stichting@nlnet.nl.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND STICHTING NLNET DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL STICHTING NLNET BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The development of Dynamically Loadable Zones (DLZ) for Bind 9 was conceived and contributed by Rob Butler.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND ROB BUTLER DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ROB BUTLER BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (c) 1987, 1990, 1993, 1994

The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

 Copyright (C) The Internet Society 2005. This version of this module is part of RFC 4178; see the RFC itself for full legal notices.

(The above copyright notice is per RFC 3978 5.6 (a), q.v.)

 Copyright (c) 2004 Masarykova universita
 (Masaryk University, Brno, Czech Republic)
 All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 1997 - 2003 Kungliga Tekniska Högskolan
(Royal Institute of Technology, Stockholm, Sweden).
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
3. Neither the name of the Institute nor the names of its contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

Copyright (c) 1998 Doug Rabson
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY

OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright ((c)) 2002, Rice University
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Rice University (RICE) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by RICE and the contributors on an "as is" basis, without any representations or warranties of any kind, express or implied including, but not limited to, representations or warranties of non-infringement, merchantability or fitness for a particular purpose. In no event shall RICE or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

Copyright (c) 1993 by Digital Equipment Corporation.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies, and that the name of Digital Equipment Corporation not be used in advertising or publicity pertaining to distribution of the document or software without specific, written prior permission.

THE SOFTWARE IS PROVIDED "AS IS" AND DIGITAL EQUIPMENT CORP. DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL DIGITAL EQUIPMENT CORPORATION BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR

PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright 2000 Aaron D. Gifford. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR(S) AND CONTRIBUTOR(S) ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR(S) OR CONTRIBUTOR(S) BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 1998 Doug Rabson.
Copyright (c) 2001 Jake Burkholder.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT

LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (C) 1995, 1996, 1997, and 1998 WIDE Project.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE PROJECT OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 1999-2000 by Nortel Networks Corporation

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND NORTEL NETWORKS DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL NORTEL NETWORKS BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (c) 2000-2002 Japan Network Information Center. All rights reserved.

By using this file, you agree to the terms and conditions set forth below.

LICENSE TERMS AND CONDITIONS

The following License Terms and Conditions apply, unless a different license is obtained from Japan Network Information Center ("JPNIC"), a Japanese association, Kokusai-Kougyou-Kanda Bldg 6F, 2-3-4 Uchi-Kanda, Chiyoda-ku, Tokyo 101-0047, Japan.

1. Use, Modification and Redistribution (including distribution of any modified or derived work) in source and/or binary forms is permitted under this License Terms and Conditions.
2. Redistribution of source code must retain the copyright notices as they appear in each source code file, this License Terms and Conditions.
3. Redistribution in binary form must reproduce the Copyright Notice, this License Terms and Conditions, in the documentation and/or other materials provided with the distribution. For the purposes of binary distribution the "Copyright Notice" refers to the following language: "Copyright (c) 2000-2002 Japan Network Information Center. All rights reserved."
4. The name of JPNIC may not be used to endorse or promote products derived from this Software without specific prior written approval of JPNIC.
5. Disclaimer/Limitation of Liability: THIS SOFTWARE IS PROVIDED BY JPNIC "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JPNIC BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Copyright (C) 2004 Nominet, Ltd.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND NOMINET DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Portions Copyright RSA Security Inc.

License to copy and use this software is granted provided that it is identified as "RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki)" in all material mentioning or referencing this software.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki)" in all material mentioning or referencing the derived work.

RSA Security Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

Copyright (c) 1996, David Mazieres <dm@uun.org>
Copyright (c) 2008, Damien Miller <djm@openbsd.org>

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (c) 2000-2001 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment:
"This product includes software developed by the OpenSSL Project

for use in the OpenSSL Toolkit. (<http://www.OpenSSL.org/>)"

4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact licensing@OpenSSL.org.
5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:
 "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.OpenSSL.org/>)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

 Copyright (c) 1995, 1997, 1998 The NetBSD Foundation, Inc.
 All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

ISC License:

Copyright © 2004-2010 by Internet Systems Consortium, Inc. ("ISC")
Copyright © 1995-2003 by Internet Software Consortium

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

1.11 binutils

1.11.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid

anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the

notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to

these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of

this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.

```
GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991
```

```
Copyright (C) 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public

Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that

they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means

all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library,

and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library

creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more

than the cost of performing this distribution.

c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying

the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Library General Public
License as published by the Free Software Foundation; either
version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Library General Public License for more details.

You should have received a copy of the GNU Library General Public
License along with this library; if not, write to the Free
Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston,
MA 02110-1301, USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!
GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest you first think carefully about whether

this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with

the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated

straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square

root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or

linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any

attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in

certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

GNU LESSER GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public

License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the

Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

1.12 bridge-utils

1.12.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy<name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.13 bzip2

1.13.1 Available under license

This program, "bzip2", the associated library "libbzip2", and all documentation, are copyright (C) 1996-2010 Julian R Seward. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Julian Seward, jseward@bzip.org
bzip2/libbzip2 version 1.0.6 of 6 September 2010

1.14 ca-certificates

1.14.1 Available under license

Format: <http://www.debian.org/doc/packaging-manuals/copyright-format/1.0/>
 Source: <http://ftp.debian.org/debian/pool/main/c/ca-certificates/>

Files: debian/*
 examples/*
 Makefile
 mozilla/*
 sbin/*

Copyright: 2003 Fumitoshi UKAI <ukai@debian.or.jp>
 2009 Philipp Kern <pkern@debian.org>
 2011 Michael Shuler <michael@pbandjelly.org>
 Various Debian Contributors

License: GPL-2+

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

.
 This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

.
 You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301, USA.

.
 On Debian GNU/Linux systems, the complete text of the GNU General Public License can be found in '/usr/share/common-licenses/GPL-2'.

Files: mozilla/certdata.txt
 mozilla/nssckbi.h

Copyright: Mozilla Contributors

Comment: Original Copyright: 1994-2000 Netscape Communications Corporation
 (certdata.txt <= CVS Revision: 1.82)

NSS no longer contains explicit copyright. Upstream indicates that "Mozilla Contributors" is an appropriate attribution for the required Copyright: field in Debian's machine-readable format.
https://bugzilla.mozilla.org/show_bug.cgi?id=850003

License: MPL-2.0

Mozilla Public License Version 2.0

=====

.
 1. Definitions

.
 1.1. "Contributor"

means each individual or legal entity that creates, contributes to the creation of, or owns Covered Software.

1.2. "Contributor Version"

means the combination of the Contributions of others (if any) used by a Contributor and that particular Contributor's Contribution.

1.3. "Contribution"

means Covered Software of a particular Contributor.

1.4. "Covered Software"

means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof.

1.5. "Incompatible With Secondary Licenses"

means

(a) that the initial Contributor has attached the notice described in Exhibit B to the Covered Software; or

(b) that the Covered Software was made available under the terms of version 1.1 or earlier of the License, but not also under the terms of a Secondary License.

1.6. "Executable Form"

means any form of the work other than Source Code Form.

1.7. "Larger Work"

means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software.

1.8. "License"

means this document.

1.9. "Licensable"

means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently, any and all of the rights conveyed by this License.

1.10. "Modifications"

means any of the following:

(a) any file in Source Code Form that results from an addition to, deletion from, or modification of the contents of Covered Software; or

(b) any new file in Source Code Form that contains any Covered Software.

1.11. "Patent Claims" of a Contributor

means any patent claim(s), including without limitation, method, process, and apparatus claims, in any patent Licensable by such

Contributor that would be infringed, but for the grant of the License, by the making, using, selling, offering for sale, having made, import, or transfer of either its Contributions or its Contributor Version.

1.12. "Secondary License"

means either the GNU General Public License, Version 2.0, the GNU Lesser General Public License, Version 2.1, the GNU Affero General Public License, Version 3.0, or any later versions of those licenses.

1.13. "Source Code Form"

means the form of the work preferred for making modifications.

1.14. "You" (or "Your")

means an individual or a legal entity exercising rights under this License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. License Grants and Conditions

2.1. Grants

Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license:

- (a) under intellectual property rights (other than patent or trademark)
 - Licensable by such Contributor to use, reproduce, make available, modify, display, perform, distribute, and otherwise exploit its Contributions, either on an unmodified basis, with Modifications, or as part of a Larger Work; and
- (b) under Patent Claims of such Contributor to make, use, sell, offer for sale, have made, import, and otherwise transfer either its Contributions or its Contributor Version.

2.2. Effective Date

The licenses granted in Section 2.1 with respect to any Contribution become effective for each Contribution on the date the Contributor first distributes such Contribution.

2.3. Limitations on Grant Scope

The licenses granted in this Section 2 are the only rights granted under this License. No additional rights or licenses will be implied from the distribution or licensing of Covered Software under this License. Notwithstanding Section 2.1(b) above, no patent license is granted by a Contributor:

- . (a) for any code that a Contributor has removed from Covered Software;
or
- . (b) for infringements caused by: (i) Your and any other third party's
modifications of Covered Software, or (ii) the combination of its
Contributions with other software (except as part of its Contributor
Version); or
- . (c) under Patent Claims infringed by Covered Software in the absence of
its Contributions.

. This License does not grant any rights in the trademarks, service marks,
or logos of any Contributor (except as may be necessary to comply with
the notice requirements in Section 3.4).

. 2.4. Subsequent Licenses

. No Contributor makes additional grants as a result of Your choice to
distribute the Covered Software under a subsequent version of this
License (see Section 10.2) or under the terms of a Secondary License (if
permitted under the terms of Section 3.3).

. 2.5. Representation

. Each Contributor represents that the Contributor believes its
Contributions are its original creation(s) or it has sufficient rights
to grant the rights to its Contributions conveyed by this License.

. 2.6. Fair Use

. This License is not intended to limit any rights You have under
applicable copyright doctrines of fair use, fair dealing, or other
equivalents.

. 2.7. Conditions

. Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted
in Section 2.1.

. 3. Responsibilities

. 3.1. Distribution of Source Form

. All distribution of Covered Software in Source Code Form, including any
Modifications that You create or to which You contribute, must be under
the terms of this License. You must inform recipients that the Source
Code Form of the Covered Software is governed by the terms of this
License, and how they can obtain a copy of this License. You may not
attempt to alter or restrict the recipients' rights in the Source Code
Form.

. 3.2. Distribution of Executable Form

If You distribute Covered Software in Executable Form then:

- (a) such Covered Software must also be made available in Source Code Form, as described in Section 3.1, and You must inform recipients of the Executable Form how they can obtain a copy of such Source Code Form by reasonable means in a timely manner, at a charge no more than the cost of distribution to the recipient; and
- (b) You may distribute such Executable Form under the terms of this License, or sublicense it under different terms, provided that the license for the Executable Form does not attempt to limit or alter the recipients' rights in the Source Code Form under this License.

3.3. Distribution of a Larger Work

You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for the Covered Software. If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, and the Covered Software is not Incompatible With Secondary Licenses, this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s).

3.4. Notices

You may not remove or alter the substance of any license notices (including copyright notices, patent notices, disclaimers of warranty, or limitations of liability) contained within the Source Code Form of the Covered Software, except that You may alter any license notices to the extent required to remedy known factual inaccuracies.

3.5. Application of Additional Terms

You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, You may do so only on Your own behalf, and not on behalf of any Contributor. You must make it absolutely clear that any such warranty, support, indemnity, or liability obligation is offered by You alone, and You hereby agree to indemnify every Contributor for any liability incurred by such Contributor as a result of warranty, support, indemnity or liability terms You offer. You may include additional disclaimers of warranty and limitations of liability specific to any jurisdiction.

4. Inability to Comply Due to Statute or Regulation

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Software due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b)

describe the limitations and the code they affect. Such description must be placed in a text file included with all distributions of the Covered Software under this License. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Termination

5.1. The rights granted under this License will terminate automatically if You fail to comply with any of its terms. However, if You become compliant, then the rights granted under this License from a particular Contributor are reinstated (a) provisionally, unless and until such Contributor explicitly and finally terminates Your grants, and (b) on an ongoing basis, if such Contributor fails to notify You of the non-compliance by some reasonable means prior to 60 days after You have come back into compliance. Moreover, Your grants from a particular Contributor are reinstated on an ongoing basis if such Contributor notifies You of the non-compliance by some reasonable means, this is the first time You have received notice of non-compliance with this License from such Contributor, and You become compliant prior to 30 days after Your receipt of the notice.

5.2. If You initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate.

5.3. In the event of termination under Sections 5.1 or 5.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or Your distributors under this License prior to termination shall survive termination.

```

*                                     *
* 6. Disclaimer of Warranty          *
* -----                          *
*                                     *
* Covered Software is provided under this License on an "as is" *
* basis, without warranty of any kind, either expressed, implied, or *
* statutory, including, without limitation, warranties that the *
* Covered Software is free of defects, merchantable, fit for a *
* particular purpose or non-infringing. The entire risk as to the *
* quality and performance of the Covered Software is with You. *
* Should any Covered Software prove defective in any respect, You *
* (not any Contributor) assume the cost of any necessary servicing, *
* repair, or correction. This disclaimer of warranty constitutes an *
* essential part of this License. No use of any Covered Software is *
* authorized under this License except under this disclaimer. *
*                                     *

```

```

*****
*
*
* 7. Limitation of Liability
* -----
*
* Under no circumstances and under no legal theory, whether tort
* (including negligence), contract, or otherwise, shall any
* Contributor, or anyone who distributes Covered Software as
* permitted above, be liable to You for any direct, indirect,
* special, incidental, or consequential damages of any character
* including, without limitation, damages for lost profits, loss of
* goodwill, work stoppage, computer failure or malfunction, or any
* and all other commercial damages or losses, even if such party
* shall have been informed of the possibility of such damages. This
* limitation of liability shall not apply to liability for death or
* personal injury resulting from such party's negligence to the
* extent applicable law prohibits such limitation. Some
* jurisdictions do not allow the exclusion or limitation of
* incidental or consequential damages, so this exclusion and
* limitation may not apply to You.
*
*****

.
8. Litigation
-----

.
Any litigation relating to this License may be brought only in the
courts of a jurisdiction where the defendant maintains its principal
place of business and such litigation shall be governed by laws of that
jurisdiction, without reference to its conflict-of-law provisions.
Nothing in this Section shall prevent a party's ability to bring
cross-claims or counter-claims.

.
9. Miscellaneous
-----

.
This License represents the complete agreement concerning the subject
matter hereof. If any provision of this License is held to be
unenforceable, such provision shall be reformed only to the extent
necessary to make it enforceable. Any law or regulation which provides
that the language of a contract shall be construed against the drafter
shall not be used to construe this License against a Contributor.

.
10. Versions of the License
-----

.
10.1. New Versions

.
Mozilla Foundation is the license steward. Except as provided in Section
10.3, no one other than the license steward has the right to modify or
publish new versions of this License. Each version will be given a
distinguishing version number.

.
10.2. Effect of New Versions

.

```


You may distribute the Covered Software under the terms of the version of the License under which You originally received the Covered Software, or under the terms of any subsequent version published by the license steward.

10.3. Modified Versions

If you create software not governed by this License, and you want to create a new license for such software, you may create and use a modified version of this License if you rename the license and remove any references to the name of the license steward (except to note that such modified license differs from this License).

10.4. Distributing Source Code Form that is Incompatible With Secondary Licenses

If You choose to distribute Source Code Form that is Incompatible With Secondary Licenses under the terms of this version of the License, the notice described in Exhibit B of this License must be attached.

Exhibit A - Source Code Form License Notice

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at <http://mozilla.org/MPL/2.0/>.

If it is not possible or desirable to put the notice in a particular file, then You may include the notice in a location (such as a LICENSE file in a relevant directory) where a recipient would be likely to look for such a notice.

You may add additional accurate notices of copyright ownership.

Exhibit B - "Incompatible With Secondary Licenses" Notice

This Source Code Form is "Incompatible With Secondary Licenses", as defined by the Mozilla Public License, v. 2.0.

1.15 chkconfig

1.15.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.16 coreutils

1.16.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <<http://www.gnu.org/licenses/>>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <<http://www.gnu.org/philosophy/why-not-lgpl.html>>. /* 'dir', 'vdir' and 'ls' directory listing programs for GNU. Copyright (C) 1985-2016 Free Software Foundation, Inc.

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>. */

```
/* If ls_mode is LS_MULTI_COL,
the multi-column format is the default regardless
of the type of output device.
This is for the 'dir' program.
```

```
If ls_mode is LS_LONG_FORMAT,
the long format is the default regardless of the
type of output device.
This is for the 'vdir' program.
```

```
If ls_mode is LS_LS,
```

```

the output format depends on whether the output
device is a terminal.
This is for the 'ls' program. */

/* Written by Richard Stallman and David MacKenzie. */

/* Color support by Peter Anvin <Peter.Anvin@linux.org> and Dennis
Flaherty <dennisf@denix.elk.miles.com> based on original patches by
Greg Lee <lee@uhunix.uhcc.hawaii.edu>. */

#include <config.h>
#include <sys/types.h>

#include <termios.h>
#if HAVE_STROPTS_H
# include <stropts.h>
#endif
#include <sys/ioctl.h>

#ifdef WINSIZE_IN_PTEM
# include <sys/stream.h>
# include <sys/ptem.h>
#endif

#include <stdio.h>
#include <assert.h>
#include <setjmp.h>
#include <pwd.h>
#include <getopt.h>
#include <signal.h>
#include <selinux/selinux.h>
#include <wchar.h>

#if HAVE_LANGINFO_CODESET
# include <langinfo.h>
#endif

/* Use SA_NOCLDSTOP as a proxy for whether the sigaction machinery is
present. */
#ifndef SA_NOCLDSTOP
# define SA_NOCLDSTOP 0
# define sigprocmask(How, Set, Oset) /* empty */
# define sigset_t int
# if ! HAVE_SIGINTERRUPT
# define siginterrupt(sig, flag) /* empty */
# endif
#endif

/* NonStop circa 2011 lacks both SA_RESTART and siginterrupt, so don't
restart syscalls after a signal handler fires. This may cause
colors to get messed up on the screen if 'ls' is interrupted, but
that's the best we can do on such a platform. */
#ifndef SA_RESTART
# define SA_RESTART 0
#endif

```

```

#include "system.h"
#include <fnmatch.h>

#include "acl.h"
#include "argmatch.h"
#include "dev-ino.h"
#include "error.h"
#include "filenamecat.h"
#include "hard-locale.h"
#include "hash.h"
#include "human.h"
#include "filemode.h"
#include "filevercmp.h"
#include "idcache.h"
#include "ls.h"
#include "mbswidth.h"
#include "mpsort.h"
#include "obstack.h"
#include "quote.h"
#include "smack.h"
#include "stat-size.h"
#include "stat-time.h"
#include "strftime.h"
#include "xdectoint.h"
#include "xstrtol.h"
#include "areadlink.h"
#include "mbsalign.h"
#include "dircolors.h"

/* Include <sys/capability.h> last to avoid a clash of <sys/types.h>
   include guards with some premature versions of libcap.
   For more details, see <http://bugzilla.redhat.com/483548>. */
#ifdef HAVE_CAP
# include <sys/capability.h>
#endif

#define PROGRAM_NAME (ls_mode == LS_LS ? "ls" \
                     : (ls_mode == LS_MULTI_COL \
                        ? "dir" : "vdir"))

#define AUTHORS \
  proper_name ("Richard M. Stallman"), \
  proper_name ("David MacKenzie")

#define obstack_chunk_alloc malloc
#define obstack_chunk_free free

/* Return an int indicating the result of comparing two integers.
   Subtracting doesn't always work, due to overflow. */
#define longdiff(a, b) ((a) < (b) ? -1 : (a) > (b))

/* Unix-based readdir implementations have historically returned a dirent.d_ino
   value that is sometimes not equal to the stat-obtained st_ino value for

```

that same entry. This error occurs for a readdir entry that refers to a mount point. readdir's error is to return the inode number of the underlying directory -- one that typically cannot be stat'ed, as long as a file system is mounted on that directory. RELIABLE_D_INO encapsulates whether we can use the more efficient approach of relying on readdir-supplied d_ino values, or whether we must incur the cost of calling stat or lstat to obtain each guaranteed-valid inode number. */

```
#ifndef READDIR_LIES_ABOUT_MOUNTPOINT_D_INO
# define READDIR_LIES_ABOUT_MOUNTPOINT_D_INO 1
#endif
```

```
#if READDIR_LIES_ABOUT_MOUNTPOINT_D_INO
# define RELIABLE_D_INO(dp) NOT_AN_INODE_NUMBER
#else
# define RELIABLE_D_INO(dp) D_INO (dp)
#endif
```

```
#if ! HAVE_STRUCT_STAT_ST_AUTHOR
# define st_author st_uid
#endif
```

```
enum filetype
{
    unknown,
    fifo,
    chardev,
    directory,
    blockdev,
    normal,
    symbolic_link,
    sock,
    whiteout,
    arg_directory
};
```

/* Display letters and indicators for each filetype.

Keep these in sync with enum filetype. */

```
static char const filetype_letter[] = "?pcdb-lswd";
```

/* Ensure that filetype and filetype_letter have the same number of elements. */

```
verify (sizeof filetype_letter - 1 == arg_directory + 1);
```

```
#define FILETYPE_INDICATORS\
{\
    C_ORPHAN, C_FIFO, C_CHR, C_DIR, C_BLK, C_FILE,\
    C_LINK, C_SOCK, C_FILE, C_DIR\
}
```

```
enum acl_type
{
    ACL_T_NONE,
    ACL_T_LSM_CONTEXT_ONLY,
    ACL_T_YES
}
```



```

};

struct fileinfo
{
    /* The file name. */
    char *name;

    /* For symbolic link, name of the file linked to, otherwise zero. */
    char *linkname;

    struct stat stat;

    enum filetype filetype;

    /* For symbolic link and long listing, st_mode of file linked to, otherwise
       zero. */
    mode_t linkmode;

    /* security context. */
    char *scontext;

    bool stat_ok;

    /* For symbolic link and color printing, true if linked-to file
       exists, otherwise false. */
    bool linkok;

    /* For long listings, true if the file has an access control list,
       or a security context. */
    enum acl_type acl_type;

    /* For color listings, true if a regular file has capability info. */
    bool has_capability;
};

#define LEN_STR_PAIR(s) sizeof(s) - 1, s

/* Null is a valid character in a color indicator (think about Epson
   printers, for example) so we have to use a length/buffer string
   type. */

struct bin_str
{
    size_t len; /* Number of bytes */
    const char *string; /* Pointer to the same */
};

#if !HAVE_TCGETPGRP
# define tcgetpgrp(Fd) 0
#endif

static size_t quote_name(FILE *out, const char *name,
                        struct quoting_options const *options,
                        size_t *width);

```

```

static char *make_link_name (char const *name, char const *linkname);
static int decode_switches (int argc, char **argv);
static bool file_ignored (char const *name);
static uintmax_t gobble_file (char const *name, enum filetype type,
                              ino_t inode, bool command_line_arg,
                              char const *dirname);
static bool print_color_indicator (const struct fileinfo *f,
                                   bool symlink_target);
static void put_indicator (const struct bin_str *ind);
static void add_ignore_pattern (const char *pattern);
static void attach (char *dest, const char *dirname, const char *name);
static void clear_files (void);
static void extract_dirs_from_files (char const *dirname,
                                     bool command_line_arg);
static void get_link_name (char const *filename, struct fileinfo *f,
                           bool command_line_arg);
static void indent (size_t from, size_t to);
static size_t calculate_columns (bool by_columns);
static void print_current_files (void);
static void print_dir (char const *name, char const *realname,
                       bool command_line_arg);
static size_t print_file_name_and_frills (const struct fileinfo *f,
                                          size_t start_col);
static void print_horizontal (void);
static int format_user_width (uid_t u);
static int format_group_width (gid_t g);
static void print_long_format (const struct fileinfo *f);
static void print_many_per_line (void);
static size_t print_name_with_quoting (const struct fileinfo *f,
                                       bool symlink_target,
                                       struct obstack *stack,
                                       size_t start_col);
static void prep_non_filename_text (void);
static bool print_type_indicator (bool stat_ok, mode_t mode,
                                  enum filetype type);
static void print_with_separator (char sep);
static void queue_directory (char const *name, char const *realname,
                             bool command_line_arg);
static void sort_files (void);
static void parse_ls_color (void);

static void getenv_quoting_style (void);

/* Initial size of hash table.
   Most hierarchies are likely to be shallower than this. */
#define INITIAL_TABLE_SIZE 30

/* The set of 'active' directories, from the current command-line argument
   to the level in the hierarchy at which files are being listed.
   A directory is represented by its device and inode numbers (struct dev_ino).
   A directory is added to this set when ls begins listing it or its
   entries, and it is removed from the set just after ls has finished
   processing it. This set is used solely to detect loops, e.g., with
   mkdir loop; cd loop; ln -s ../loop sub; ls -RL */
static Hash_table *active_dir_set;

```

```

#define LOOP_DETECT (!active_dir_set)

/* The table of files in the current directory:

'cwd_file' points to a vector of 'struct fileinfo', one per file.
'cwd_n_alloc' is the number of elements space has been allocated for.
'cwd_n_used' is the number actually in use. */

/* Address of block containing the files that are described. */
static struct fileinfo *cwd_file;

/* Length of block that 'cwd_file' points to, measured in files. */
static size_t cwd_n_alloc;

/* Index of first unused slot in 'cwd_file'. */
static size_t cwd_n_used;

/* Vector of pointers to files, in proper sorted order, and the number
   of entries allocated for it. */
static void **sorted_file;
static size_t sorted_file_alloc;

/* When true, in a color listing, color each symlink name according to the
   type of file it points to. Otherwise, color them according to the 'ln'
   directive in LS_COLORS. Dangling (orphan) symlinks are treated specially,
   regardless. This is set when 'ln=target' appears in LS_COLORS. */

static bool color_symlink_as_referent;

/* mode of appropriate file for colorization */
#define FILE_OR_LINK_MODE(File) \
    ((color_symlink_as_referent && (File)->linkok) \
     ? (File)->linkmode : (File)->stat.st_mode)

/* Record of one pending directory waiting to be listed. */

struct pending
{
    char *name;
    /* If the directory is actually the file pointed to by a symbolic link we
       were told to list, 'realname' will contain the name of the symbolic
       link, otherwise zero. */
    char *realname;
    bool command_line_arg;
    struct pending *next;
};

static struct pending *pending_dirs;

/* Current time in seconds and nanoseconds since 1970, updated as
   needed when deciding whether a file is recent. */

```

```

static struct timespec current_time;

static bool print_scontext;
static char UNKNOWN_SECURITY_CONTEXT[] = "?";

/* Whether any of the files has an ACL. This affects the width of the
   mode column. */

static bool any_has_acl;

/* The number of columns to use for columns containing inode numbers,
   block sizes, link counts, owners, groups, authors, major device
   numbers, minor device numbers, and file sizes, respectively. */

static int inode_number_width;
static int block_size_width;
static int nlink_width;
static int sccontext_width;
static int owner_width;
static int group_width;
static int author_width;
static int major_device_number_width;
static int minor_device_number_width;
static int file_size_width;

/* Option flags */

/* long_format for lots of info, one per line.
   one_per_line for just names, one per line.
   many_per_line for just names, many per line, sorted vertically.
   horizontal for just names, many per line, sorted horizontally.
   with_commas for just names, many per line, separated by commas.

   -l (and other options that imply -l), -l, -C, -x and -m control
   this parameter. */

enum format
{
    long_format, /* -l and other options that imply -l */
    one_per_line, /* -l */
    many_per_line, /* -C */
    horizontal, /* -x */
    with_commas, /* -m */
};

static enum format format;

/* 'full-iso' uses full ISO-style dates and times. 'long-iso' uses longer
   ISO-style time stamps, though shorter than 'full-iso'. 'iso' uses shorter
   ISO-style time stamps. 'locale' uses locale-dependent time stamps. */
enum time_style
{
    full_iso_time_style, /* --time-style=full-iso */
    long_iso_time_style, /* --time-style=long-iso */
    iso_time_style, /* --time-style=iso */

```

```

    locale_time_style/* --time-style=locale */
};

static char const *const time_style_args[] =
{
    "full-iso", "long-iso", "iso", "locale", NULL
};
static enum time_style const time_style_types[] =
{
    full_iso_time_style, long_iso_time_style, iso_time_style,
    locale_time_style
};
ARGMATCH_VERIFY (time_style_args, time_style_types);

/* Type of time to print or sort by.  Controlled by -c and -u.
   The values of each item of this enum are important since they are
   used as indices in the sort functions array (see sort_files()). */

enum time_type
{
    time_mtime/* default */
    time_ctime/* -c */
    time_atime/* -u */
    time_numtypes/* the number of elements of this enum */
};

static enum time_type time_type;

/* The file characteristic to sort by.  Controlled by -t, -S, -U, -X, -v.
   The values of each item of this enum are important since they are
   used as indices in the sort functions array (see sort_files()). */

enum sort_type
{
    sort_none = -1/* -U */
    sort_name/* default */
    sort_extension/* -X */
    sort_size/* -S */
    sort_version/* -v */
    sort_time/* -t */
    sort_numtypes/* the number of elements of this enum */
};

static enum sort_type sort_type;

/* Direction of sort.
   false means highest first if numeric,
   lowest first if alphabetic;
   these are the defaults.
   true means the opposite order in each case.  -r */

static bool sort_reverse;

/* True means to display owner information.  -g turns this off. */

```

```

static bool print_owner = true;

/* True means to display author information. */

static bool print_author;

/* True means to display group information. -G and -o turn this off. */

static bool print_group = true;

/* True means print the user and group id's as numbers rather
   than as names. -n */

static bool numeric_ids;

/* True means mention the size in blocks of each file. -s */

static bool print_block_size;

/* Human-readable options for output, when printing block counts. */
static int human_output_opts;

/* The units to use when printing block counts. */
static uintmax_t output_block_size;

/* Likewise, but for file sizes. */
static int file_human_output_opts;
static uintmax_t file_output_block_size = 1;

/* Follow the output with a special string. Using this format,
   Emacs' dire mode starts up twice as fast, and can handle all
   strange characters in file names. */
static bool dire;

/* 'none' means don't mention the type of files.
   'slash' means mention directories only, with a '/'.
   'file_type' means mention file types.
   'classify' means mention file types and mark executables.

   Controlled by -F, -p, and --indicator-style. */

enum indicator_style
{
    none, /* --indicator-style=none */
    slash, /* -p, --indicator-style=slash */
    file_type, /* --indicator-style=file-type */
    classify, /* -F, --indicator-style=classify */
};

static enum indicator_style indicator_style;

/* Names of indicator styles. */
static char const *const indicator_style_args[] =
{

```

```

    "none", "slash", "file-type", "classify", NULL
};
static enum indicator_style const indicator_style_types[] =
{
    none, slash, file_type, classify
};
ARGMATCH_VERIFY (indicator_style_args, indicator_style_types);

/* True means use colors to mark types. Also define the different
   colors as well as the stuff for the LS_COLORS environment variable.
   The LS_COLORS variable is now in a termcap-like format. */

static bool print_with_color;

/* Whether we used any colors in the output so far. If so, we will
   need to restore the default color later. If not, we will need to
   call prep_non_filename_text before using color for the first time. */

static bool used_color = false;

enum color_type
{
    color_never /* 0: default or --color=never */
    color_always /* 1: --color=always */
    color_if_tty /* 2: --color=tty */
};

enum Dereference_symlink
{
    DEREf_UNDEFINED = 1,
    DEREf_NEVER,
    DEREf_COMMAND_LINE_ARGUMENTS /* -H */
    DEREf_COMMAND_LINE_SYMLINK_TO_DIR /* the default, in certain cases */
    DEREf_ALWAYS /* -L */
};

enum indicator_no
{
    C_LEFT, C_RIGHT, C_END, C_RESET, C_NORM, C_FILE, C_DIR, C_LINK,
    C_FIFO, C_SOCKET,
    C_BLK, C_CHR, C_MISSING, C_ORPHAN, C_EXEC, C_DOOR, C_SETUID, C_SETGID,
    C_STICKY, C_OTHER_WRITABLE, C_STICKY_OTHER_WRITABLE, C_CAP, C_MULTIHARDLINK,
    C_CLR_TO_EOL
};

static const char *const indicator_name[] =
{
    "lc", "rc", "ec", "rs", "no", "fi", "di", "ln", "pi", "so",
    "bd", "cd", "mi", "or", "ex", "do", "su", "sg", "st",
    "ow", "tw", "ca", "mh", "cl", NULL
};

struct color_ext_type
{

```

```

    struct bin_str ext; /* The extension we're looking for */
    struct bin_str seq; /* The sequence to output when we do */
    struct color_ext_type *next; /* Next in list */
};

static struct bin_str color_indicator[] =
{
    { LEN_STR_PAIR ("\033[") }, /* lc: Left of color sequence */
    { LEN_STR_PAIR ("m") }, /* rc: Right of color sequence */
    { 0, NULL }, /* ec: End color (replaces lc+rs+rc) */
    { LEN_STR_PAIR ("0") }, /* rs: Reset to ordinary colors */
    { 0, NULL }, /* no: Normal */
    { 0, NULL }, /* fi: File: default */
    { LEN_STR_PAIR ("01;34") }, /* di: Directory: bright blue */
    { LEN_STR_PAIR ("01;36") }, /* ln: Symlink: bright cyan */
    { LEN_STR_PAIR ("33") }, /* pi: Pipe: yellow/brown */
    { LEN_STR_PAIR ("01;35") }, /* so: Socket: bright magenta */
    { LEN_STR_PAIR ("01;33") }, /* bd: Block device: bright yellow */
    { LEN_STR_PAIR ("01;33") }, /* cd: Char device: bright yellow */
    { 0, NULL }, /* mi: Missing file: undefined */
    { 0, NULL }, /* or: Orphaned symlink: undefined */
    { LEN_STR_PAIR ("01;32") }, /* ex: Executable: bright green */
    { LEN_STR_PAIR ("01;35") }, /* do: Door: bright magenta */
    { LEN_STR_PAIR ("37;41") }, /* su: setuid: white on red */
    { LEN_STR_PAIR ("30;43") }, /* sg: setgid: black on yellow */
    { LEN_STR_PAIR ("37;44") }, /* st: sticky: black on blue */
    { LEN_STR_PAIR ("34;42") }, /* ow: other-writable: blue on green */
    { LEN_STR_PAIR ("30;42") }, /* tw: ow w/ sticky: black on green */
    { LEN_STR_PAIR ("30;41") }, /* ca: black on red */
    { 0, NULL }, /* mh: disabled by default */
    { LEN_STR_PAIR ("\033[K") }, /* cl: clear to end of line */
};

/* FIXME: comment */
static struct color_ext_type *color_ext_list = NULL;

/* Buffer for color sequences */
static char *color_buf;

/* True means to check for orphaned symbolic link, for displaying
   colors. */

static bool check_symlink_color;

/* True means mention the inode number of each file. -i */

static bool print_inode;

/* What to do with symbolic links. Affected by -d, -F, -H, -l (and
   other options that imply -l), and -L. */

static enum Dereference_symlink dereference;

/* True means when a directory is found, display info on its
   contents. -R */

```



```

static bool recursive;

/* True means when an argument is a directory name, display info
   on it itself. -d */

static bool immediate_dirs;

/* True means that directories are grouped before files. */

static bool directories_first;

/* Which files to ignore. */

static enum
{
    /* Ignore files whose names start with '.', and files specified by
       --hide and --ignore. */
    IGNORE_DEFAULT,

    /* Ignore '.', '..', and files specified by --ignore. */
    IGNORE_DOT_AND_DOTDOT,

    /* Ignore only files specified by --ignore. */
    IGNORE_MINIMAL
} ignore_mode;

/* A linked list of shell-style globbing patterns. If a non-argument
   file name matches any of these patterns, it is ignored.
   Controlled by -I. Multiple -I options accumulate.
   The -B option adds '*~' and '*.~' to this list. */

struct ignore_pattern
{
    const char *pattern;
    struct ignore_pattern *next;
};

static struct ignore_pattern *ignore_patterns;

/* Similar to IGNORE_PATTERNS, except that -a or -A causes this
   variable itself to be ignored. */
static struct ignore_pattern *hide_patterns;

/* True means output nongraphic chars in file names as '?'.
   (-q, --hide-control-chars)
   qmark_funny_chars and the quoting style (-Q, --quoting-style=WORD) are
   independent. The algorithm is: first, obey the quoting style to get a
   string representing the file name; then, if qmark_funny_chars is set,
   replace all nonprintable chars in that string with '?'. It's necessary
   to replace nonprintable chars even in quoted strings, because we don't
   want to mess up the terminal if control chars get sent to it, and some
   quoting methods pass through control chars as-is. */
static bool qmark_funny_chars;

```

```

/* Quoting options for file and dir name output. */

static struct quoting_options *filename_quoting_options;
static struct quoting_options *dirname_quoting_options;

/* The number of chars per hardware tab stop. Setting this to zero
   inhibits the use of TAB characters for separating columns. -T */
static size_t tabsize;

/* True means print each directory name before listing it. */

static bool print_dir_name;

/* The line length to use for breaking lines in many-per-line format.
   Can be set with -w. */

static size_t line_length;

/* The local time zone rules, as per the TZ environment variable. */

static timezone_t localtime;

/* If true, the file listing format requires that stat be called on
   each file. */

static bool format_needs_stat;

/* Similar to 'format_needs_stat', but set if only the file type is
   needed. */

static bool format_needs_type;

/* An arbitrary limit on the number of bytes in a printed time stamp.
   This is set to a relatively small value to avoid the need to worry
   about denial-of-service attacks on servers that run "ls" on behalf
   of remote clients. 1000 bytes should be enough for any practical
   time stamp format. */

enum { TIME_STAMP_LEN_MAXIMUM = MAX (1000, INT_STRLEN_BOUND (time_t)) };

/* strftime formats for non-recent and recent files, respectively, in
   -l output. */

static char const *long_time_format[2] =
{
    /* strftime format for non-recent files (older than 6 months), in
       -l output. This should contain the year, month and day (at
       least), in an order that is understood by people in your
       locale's territory. Please try to keep the number of used
       screen columns small, because many people work in windows with
       only 80 columns. But make this as wide as the other string
       below, for recent files. */
    /* TRANSLATORS: ls output needs to be aligned for ease of reading,
       so be wary of using variable width fields from the locale.

```

```

    Note %b is handled specially by ls and aligned correctly.
    Note also that specifying a width as in %5b is erroneous as strftime
    will count bytes rather than characters in multibyte locales. */
    N_("%b %e %Y"),
    /* strftime format for recent files (younger than 6 months), in -l
    output. This should contain the month, day and time (at
    least), in an order that is understood by people in your
    locale's territory. Please try to keep the number of used
    screen columns small, because many people work in windows with
    only 80 columns. But make this as wide as the other string
    above, for non-recent files. */
    /* TRANSLATORS: ls output needs to be aligned for ease of reading,
    so be wary of using variable width fields from the locale.
    Note %b is handled specially by ls and aligned correctly.
    Note also that specifying a width as in %5b is erroneous as strftime
    will count bytes rather than characters in multibyte locales. */
    N_("%b %e %H:%M")
};

/* The set of signals that are caught. */

static sigset_t caught_signals;

/* If nonzero, the value of the pending fatal signal. */

static sig_atomic_t volatile interrupt_signal;

/* A count of the number of pending stop signals that have been received. */

static sig_atomic_t volatile stop_signal_count;

/* Desired exit status. */

static int exit_status;

/* Exit statuses. */
enum
{
    /* "ls" had a minor problem. E.g., while processing a directory,
    ls obtained the name of an entry via readdir, yet was later
    unable to stat that name. This happens when listing a directory
    in which entries are actively being removed or renamed. */
    LS_MINOR_PROBLEM = 1,

    /* "ls" had more serious trouble (e.g., memory exhausted, invalid
    option or failure to stat a command line argument. */
    LS_FAILURE = 2
};

/* For long options that have no equivalent short option, use a
non-character as a pseudo short option, starting with CHAR_MAX + 1. */
enum
{
    AUTHOR_OPTION = CHAR_MAX + 1,

```

```

BLOCK_SIZE_OPTION,
COLOR_OPTION,
DEREFERENCE_COMMAND_LINE_SYMLINK_TO_DIR_OPTION,
FILE_TYPE_INDICATOR_OPTION,
FORMAT_OPTION,
FULL_TIME_OPTION,
GROUP_DIRECTORIES_FIRST_OPTION,
HIDE_OPTION,
INDICATOR_STYLE_OPTION,
QUOTING_STYLE_OPTION,
SHOW_CONTROL_CHARS_OPTION,
SI_OPTION,
SORT_OPTION,
TIME_OPTION,
TIME_STYLE_OPTION
};

static struct option const long_options[] =
{
    {"all", no_argument, NULL, 'a'},
    {"escape", no_argument, NULL, 'b'},
    {"directory", no_argument, NULL, 'd'},
    {"dired", no_argument, NULL, 'D'},
    {"full-time", no_argument, NULL, FULL_TIME_OPTION},
    {"group-directories-first", no_argument, NULL,
    GROUP_DIRECTORIES_FIRST_OPTION},
    {"human-readable", no_argument, NULL, 'h'},
    {"inode", no_argument, NULL, 'i'},
    {"kibibytes", no_argument, NULL, 'k'},
    {"numeric-uid-gid", no_argument, NULL, 'n'},
    {"no-group", no_argument, NULL, 'G'},
    {"hide-control-chars", no_argument, NULL, 'q'},
    {"reverse", no_argument, NULL, 'r'},
    {"size", no_argument, NULL, 's'},
    {"width", required_argument, NULL, 'w'},
    {"almost-all", no_argument, NULL, 'A'},
    {"ignore-backups", no_argument, NULL, 'B'},
    {"classify", no_argument, NULL, 'F'},
    {"file-type", no_argument, NULL, FILE_TYPE_INDICATOR_OPTION},
    {"si", no_argument, NULL, SI_OPTION},
    {"dereference-command-line", no_argument, NULL, 'H'},
    {"dereference-command-line-symlink-to-dir", no_argument, NULL,
    DEREFERENCE_COMMAND_LINE_SYMLINK_TO_DIR_OPTION},
    {"hide", required_argument, NULL, HIDE_OPTION},
    {"ignore", required_argument, NULL, 'I'},
    {"indicator-style", required_argument, NULL, INDICATOR_STYLE_OPTION},
    {"dereference", no_argument, NULL, 'L'},
    {"literal", no_argument, NULL, 'N'},
    {"quote-name", no_argument, NULL, 'Q'},
    {"quoting-style", required_argument, NULL, QUOTING_STYLE_OPTION},
    {"recursive", no_argument, NULL, 'R'},
    {"format", required_argument, NULL, FORMAT_OPTION},
    {"show-control-chars", no_argument, NULL, SHOW_CONTROL_CHARS_OPTION},
    {"sort", required_argument, NULL, SORT_OPTION},
    {"tabsize", required_argument, NULL, 'T'},

```

```

    {"time", required_argument, NULL, TIME_OPTION},
    {"time-style", required_argument, NULL, TIME_STYLE_OPTION},
    {"color", optional_argument, NULL, COLOR_OPTION},
    {"block-size", required_argument, NULL, BLOCK_SIZE_OPTION},
    {"context", no_argument, 0, 'Z'},
    {"author", no_argument, NULL, AUTHOR_OPTION},
    {GETOPT_HELP_OPTION_DECL},
    {GETOPT_VERSION_OPTION_DECL},
    {NULL, 0, NULL, 0}
};

static char const *const format_args[] =
{
    "verbose", "long", "commas", "horizontal", "across",
    "vertical", "single-column", NULL
};
static enum format const format_types[] =
{
    long_format, long_format, with_commas, horizontal, horizontal,
    many_per_line, one_per_line
};
ARGMATCH_VERIFY (format_args, format_types);

static char const *const sort_args[] =
{
    "none", "time", "size", "extension", "version", NULL
};
static enum sort_type const sort_types[] =
{
    sort_none, sort_time, sort_size, sort_extension, sort_version
};
ARGMATCH_VERIFY (sort_args, sort_types);

static char const *const time_args[] =
{
    "atime", "access", "use", "ctime", "status", NULL
};
static enum time_type const time_types[] =
{
    time_atime, time_atime, time_atime, time_ctime, time_ctime
};
ARGMATCH_VERIFY (time_args, time_types);

static char const *const color_args[] =
{
    /* force and none are for compatibility with another color-ls version */
    "always", "yes", "force",
    "never", "no", "none",
    "auto", "tty", "if-tty", NULL
};
static enum color_type const color_types[] =
{
    color_always, color_always, color_always,
    color_never, color_never, color_never,

```

```

    color_if_tty, color_if_tty, color_if_tty
};
ARGMATCH_VERIFY (color_args, color_types);

/* Information about filling a column. */
struct column_info
{
    bool valid_len;
    size_t line_len;
    size_t *col_arr;
};

/* Array with information about column filledness. */
static struct column_info *column_info;

/* Maximum number of columns ever possible for this display. */
static size_t max_idx;

/* The minimum width of a column is 3: 1 character for the name and 2
   for the separating white space. */
#define MIN_COLUMN_WIDTH3

/* This zero-based index is used solely with the --dired option.
   When that option is in effect, this counter is incremented for each
   byte of output generated by this program so that the beginning
   and ending indices (in that output) of every file name can be recorded
   and later output themselves. */
static size_t dired_pos;

#define Dired_PUTCHAR(c) do {putchar ((c)); ++dired_pos;} while (0)

/* Write S to STREAM and increment Dired_POS by S_LEN. */
#define Dired_FPUTS(s, stream, s_len) \
    do {fputs (s, stream); dired_pos += s_len;} while (0)

/* Like Dired_FPUTS, but for use when S is a literal string. */
#define Dired_FPUTS_LITERAL(s, stream) \
    do {fputs (s, stream); dired_pos += sizeof (s) - 1;} while (0)

#define Dired_INDENT() \
    do \
    { \
        if (dired) \
            Dired_FPUTS_LITERAL (" ", stdout); \
    } \
    while (0)

/* With --dired, store pairs of beginning and ending indices of file names. */
static struct obstack dired_obstack;

/* With --dired, store pairs of beginning and ending indices of any
   directory names that appear as headers (just before 'total' line)
   for lists of directory entries. Such directory names are seen when
   listing hierarchies using -R and when a directory is listed with at

```

```

    least one other command line argument. */
static struct obstack subdired_obstack;

/* Save the current index on the specified obstack, OBS. */
#define PUSH_CURRENT_DIRED_POS(obs)\
do\
{\
    if (dired)\
        obstack_grow (obs, &dired_pos, sizeof (dired_pos));\
}\
while (0)

/* With -R, this stack is used to help detect directory cycles.
   The device/inode pairs on this stack mirror the pairs in the
   active_dir_set hash table. */
static struct obstack dev_ino_obstack;

/* Push a pair onto the device/inode stack. */
static void
dev_ino_push (dev_t dev, ino_t ino)
{
    void *vdi;
    struct dev_ino *di;
    int dev_ino_size = sizeof *di;
    obstack_blank (&dev_ino_obstack, dev_ino_size);
    vdi = obstack_next_free (&dev_ino_obstack);
    di = vdi;
    di--;
    di->st_dev = dev;
    di->st_ino = ino;
}

/* Pop a dev/ino struct off the global dev_ino_obstack
   and return that struct. */
static struct dev_ino
dev_ino_pop (void)
{
    void *vdi;
    struct dev_ino *di;
    int dev_ino_size = sizeof *di;
    assert (dev_ino_size <= obstack_object_size (&dev_ino_obstack));
    obstack_blank_fast (&dev_ino_obstack, -dev_ino_size);
    vdi = obstack_next_free (&dev_ino_obstack);
    di = vdi;
    return *di;
}

/* Note the use commented out below:
#define ASSERT_MATCHING_DEV_INO(Name, Di)\
do\
{\
    struct stat sb;\
    assert (Name);\
    assert (0 <= stat (Name, &sb));\
}

```

```

    assert (sb.st_dev == Di.st_dev);\
    assert (sb.st_ino == Di.st_ino);\
}\
while (0)
*/

/* Write to standard output PREFIX, followed by the quoting style and
   a space-separated list of the integers stored in OS all on one line. */

static void
dired_dump_obstack (const char *prefix, struct obstack *os)
{
    size_t n_pos;

    n_pos = obstack_object_size (os) / sizeof (dired_pos);
    if (n_pos > 0)
    {
        size_t i;
        size_t *pos;

        pos = (size_t *) obstack_finish (os);
        fputs (prefix, stdout);
        for (i = 0; i < n_pos; i++)
            printf ("%lu", (unsigned long int) pos[i]);
        putchar ('\n');
    }
}

/* Read the abbreviated month names from the locale, to align them
   and to determine the max width of the field and to truncate names
   greater than our max allowed.
   Note even though this handles multibyte locales correctly
   it's not restricted to them as single byte locales can have
   variable width abbreviated months and also precomputing/caching
   the names was seen to increase the performance of ls significantly. */

/* max number of display cells to use */
enum { MAX_MON_WIDTH = 5 };
/* In the unlikely event that the abmon[] storage is not big enough
   an error message will be displayed, and we revert to using
   unmodified abbreviated month names from the locale database. */
static char abmon[12][MAX_MON_WIDTH * 2 * MB_LEN_MAX + 1];
/* minimum width needed to align %b, 0 => don't use precomputed values. */
static size_t required_mon_width;

static size_t
abmon_init (void)
{
#ifdef HAVE_NL_LANGINFO
    required_mon_width = MAX_MON_WIDTH;
    size_t curr_max_width;
    do
    {
        curr_max_width = required_mon_width;
        required_mon_width = 0;

```



```

    for (int i = 0; i < 12; i++)
    {
        size_t width = curr_max_width;

        size_t req = mbsalign (nl_langinfo (ABMON_1 + i),
                                abmon[i], sizeof (abmon[i]),
                                &width, MBS_ALIGN_LEFT, 0);

        if (req == (size_t) -1 || req >= sizeof (abmon[i]))
        {
            required_mon_width = 0; /* ignore precomputed strings. */
            return required_mon_width;
        }

        required_mon_width = MAX (required_mon_width, width);
    }
    while (curr_max_width > required_mon_width);
#endif

    return required_mon_width;
}

static size_t
dev_ino_hash (void const *x, size_t table_size)
{
    struct dev_ino const *p = x;
    return (uintmax_t) p->st_ino % table_size;
}

static bool
dev_ino_compare (void const *x, void const *y)
{
    struct dev_ino const *a = x;
    struct dev_ino const *b = y;
    return SAME_INODE (*a, *b) ? true : false;
}

static void
dev_ino_free (void *x)
{
    free (x);
}

/* Add the device/inode pair (P->st_dev/P->st_ino) to the set of
   active directories. Return true if there is already a matching
   entry in the table. */

static bool
visit_dir (dev_t dev, ino_t ino)
{
    struct dev_ino *ent;
    struct dev_ino *ent_from_table;
    bool found_match;

```

```

ent = xmalloc (sizeof *ent);
ent->st_ino = ino;
ent->st_dev = dev;

/* Attempt to insert this entry into the table. */
ent_from_table = hash_insert (active_dir_set, ent);

if (ent_from_table == NULL)
{
    /* Insertion failed due to lack of memory. */
    xalloc_die ();
}

found_match = (ent_from_table != ent);

if (found_match)
{
    /* ent was not inserted, so free it. */
    free (ent);
}

return found_match;
}

static void
free_pending_ent (struct pending *p)
{
    free (p->name);
    free (p->realname);
    free (p);
}

static bool
is_colored (enum indicator_no type)
{
    size_t len = color_indicator[type].len;
    char const *s = color_indicator[type].string;
    return ! (len == 0
        || (len == 1 && STRNCMP_LIT (s, "0") == 0)
        || (len == 2 && STRNCMP_LIT (s, "00") == 0));
}

static void
restore_default_color (void)
{
    put_indicator (&color_indicator[C_LEFT]);
    put_indicator (&color_indicator[C_RIGHT]);
}

static void
set_normal_color (void)
{
    if (print_with_color && is_colored (C_NORM))
    {

```

```
    put_indicator (&color_indicator[C_LEFT]);
    put_indicator (&color_indicator[C_NORM]);
    put_indicator (&color_indicator[C_RIGHT]);
}
}

/* An ordinary signal was received; arrange for the program to exit. */

static void
sighandler (int sig)
{
    if (! SA_NOCLDSTOP)
        signal (sig, SIG_IGN);
    if (! interrupt_signal)
        interrupt_signal = sig;
}

/* A SIGTSTP was received; arrange for the program to suspend itself. */

static void
stophandler (int sig)
{
    if (! SA_NOCLDSTOP)
        signal (sig, stophandler);
    if (! interrupt_signal)
        stop_signal_count++;
}

/* Process any pending signals. If signals are caught, this function
should be called periodically. Ideally there should never be an
unbounded amount of time when signals are not being processed.
Signal handling can restore the default colors, so callers must
immediately change colors after invoking this function. */

static void
process_signals (void)
{
    while (interrupt_signal || stop_signal_count)
    {
        int sig;
        int stops;
        sigset_t oldset;

        if (used_color)
            restore_default_color ();
        fflush (stdout);

        sigprocmask (SIG_BLOCK, &caught_signals, &oldset);

        /* Reload interrupt_signal and stop_signal_count, in case a new
        signal was handled before sigprocmask took effect. */
        sig = interrupt_signal;
        stops = stop_signal_count;
    }
}
```

```

/* SIGTSTP is special, since the application can receive that signal
   more than once. In this case, don't set the signal handler to the
   default. Instead, just raise the uncatchable SIGSTOP. */
if (stops)
{
    stop_signal_count = stops - 1;
    sig = SIGSTOP;
}
else
    signal (sig, SIG_DFL);

/* Exit or suspend the program. */
raise (sig);
sigprocmask (SIG_SETMASK, &oldset, NULL);

/* If execution reaches here, then the program has been
   continued (after being suspended). */
}
}

int
main (int argc, char **argv)
{
    int i;
    struct pending *thispend;
    int n_files;

    /* The signals that are trapped, and the number of such signals. */
    static int const sig[] =
    {
        /* This one is handled specially. */
        SIGTSTP,

        /* The usual suspects. */
        SIGALRM, SIGHUP, SIGINT, SIGPIPE, SIGQUIT, SIGTERM,
#ifdef SIGPOLL
        SIGPOLL,
#endif
#ifdef SIGPROF
        SIGPROF,
#endif
#ifdef SIGVTALRM
        SIGVTALRM,
#endif
#ifdef SIGXCPU
        SIGXCPU,
#endif
#ifdef SIGXFSZ
        SIGXFSZ,
#endif
    };
    enum { nsigs = ARRAY_CARDINALITY (sig) };

#ifdef ! SA_NOCLDSTOP
    bool caught_sig[nsigs];

```

```

#endif

initialize_main (&argc, &argv);
set_program_name (argv[0]);
setlocale (LC_ALL, "");
bindtextdomain (PACKAGE, LOCALEDIR);
textdomain (PACKAGE);

initialize_exit_failure (LS_FAILURE);
atexit (close_stdout);

assert (ARRAY_CARDINALITY (color_indicator) + 1
        == ARRAY_CARDINALITY (indicator_name));

exit_status = EXIT_SUCCESS;
print_dir_name = true;
pending_dirs = NULL;

current_time.tv_sec = TYPE_MINIMUM (time_t);
current_time.tv_nsec = -1;

i = decode_switches (argc, argv);

if (print_with_color)
  parse_ls_color ();

/* Test print_with_color again, because the call to parse_ls_color
   may have just reset it -- e.g., if LS_COLORS is invalid. */
if (print_with_color)
{
  /* Avoid following symbolic links when possible. */
  if (is_colored (C_ORPHAN)
      || (is_colored (C_EXEC) && color_symlink_as_referent)
      || (is_colored (C_MISSING) && format == long_format))
    check_symlink_color = true;

  /* If the standard output is a controlling terminal, watch out
     for signals, so that the colors can be restored to the
     default state if "ls" is suspended or interrupted. */

  if (0 <= tcgetpgrp (STDOUT_FILENO))
  {
    int j;
    #if SA_NOCLDSTOP
      struct sigaction act;

      sigemptyset (&caught_signals);
      for (j = 0; j < nsigs; j++)
      {
        sigaction (sig[j], NULL, &act);
        if (act.sa_handler != SIG_IGN)
          sigaddset (&caught_signals, sig[j]);
      }
    #endif
  }
}

```

```

    act.sa_mask = caught_signals;
    act.sa_flags = SA_RESTART;

    for (j = 0; j < nsigs; j++)
        if (sigismember (&caught_signals, sig[j]))
        {
            act.sa_handler = sig[j] == SIGTSTP ? stophandler : sighandler;
            sigaction (sig[j], &act, NULL);
        }
    #else
        for (j = 0; j < nsigs; j++)
        {
            caught_sig[j] = (signal (sig[j], SIG_IGN) != SIG_IGN);
            if (caught_sig[j])
            {
                signal (sig[j], sig[j] == SIGTSTP ? stophandler : sighandler);
                siginterrupt (sig[j], 0);
            }
        }
    #endif
}

if (dereference == Deref_UNDEFINED)
    dereference = ((immediate_dirs
        || indicator_style == classify
        || format == long_format)
        ? Deref_NEVER
        : Deref_COMMAND_LINE_SYMLINK_TO_DIR);

/* When using -R, initialize a data structure we'll use to
   detect any directory cycles. */
if (recursive)
{
    active_dir_set = hash_initialize (INITIAL_TABLE_SIZE, NULL,
                                    dev_ino_hash,
                                    dev_ino_compare,
                                    dev_ino_free);
    if (active_dir_set == NULL)
        xalloc_die ();

    obstack_init (&dev_ino_obstack);
}

localtz = tzalloc (getenv ("TZ"));

format_needs_stat = sort_type == sort_time || sort_type == sort_size
    || format == long_format
    || print_scontext
    || print_block_size;
format_needs_type = (! format_needs_stat
    && (recursive
        || print_with_color
        || indicator_style != none
        || directories_first));

```

```

if (dired)
{
    obstack_init (&dired_obstack);
    obstack_init (&subdired_obstack);
}

cwd_n_alloc = 100;
cwd_file = xnmalloc (cwd_n_alloc, sizeof *cwd_file);
cwd_n_used = 0;

clear_files ();

n_files = argc - i;

if (n_files <= 0)
{
    if (immediate_dirs)
        gobble_file (".", directory, NOT_AN_INODE_NUMBER, true, "");
    else
        queue_directory (".", NULL, true);
}
else
do
    gobble_file (argv[i++], unknown, NOT_AN_INODE_NUMBER, true, "");
while (i < argc);

if (cwd_n_used)
{
    sort_files ();
    if (!immediate_dirs)
        extract_dirs_from_files (NULL, true);
    /* 'cwd_n_used' might be zero now. */
}

/* In the following if/else blocks, it is sufficient to test 'pending_dirs'
   (and not pending_dirs->name) because there may be no markers in the queue
   at this point. A marker may be enqueued when extract_dirs_from_files is
   called with a non-empty string or via print_dir. */
if (cwd_n_used)
{
    print_current_files ();
    if (pending_dirs)
        DIREN_PUTCHAR ('\n');
}
else if (n_files <= 1 && pending_dirs && pending_dirs->next == 0)
    print_dir_name = false;

while (pending_dirs)
{
    thispend = pending_dirs;
    pending_dirs = pending_dirs->next;

    if (LOOP_DETECT)

```

```

    {
        if (thispend->name == NULL)
        {
            /* thispend->name == NULL means this is a marker entry
               indicating we've finished processing the directory.
               Use its dev/ino numbers to remove the corresponding
               entry from the active_dir_set hash table. */
            struct dev_ino di = dev_ino_pop ();
            struct dev_ino *found = hash_delete (active_dir_set, &di);
            /* ASSERT_MATCHING_DEV_INO (thispend->realname, di); */
            assert (found);
            dev_ino_free (found);
            free_pending_ent (thispend);
            continue;
        }
    }

    print_dir (thispend->name, thispend->realname,
               thispend->command_line_arg);

    free_pending_ent (thispend);
    print_dir_name = true;
}

if (print_with_color)
{
    int j;

    if (used_color)
    {
        /* Skip the restore when it would be a no-op, i.e.,
           when left is "\033[" and right is "m". */
        if (!(color_indicator[C_LEFT].len == 2
              && memcmp (color_indicator[C_LEFT].string, "\033[", 2) == 0
              && color_indicator[C_RIGHT].len == 1
              && color_indicator[C_RIGHT].string[0] == 'm'))
            restore_default_color ();
    }
    fflush (stdout);

    /* Restore the default signal handling. */
    #if SA_NOCLDSTOP
    for (j = 0; j < nsigs; j++)
        if (sigismember (&caught_signals, sig[j]))
            signal (sig[j], SIG_DFL);
    #else
    for (j = 0; j < nsigs; j++)
        if (caught_sig[j])
            signal (sig[j], SIG_DFL);
    #endif

    /* Act on any signals that arrived before the default was restored.
       This can process signals out of order, but there doesn't seem to
       be an easy way to do them in order, and the order isn't that
       important anyway. */

```



```

        for (j = stop_signal_count; j; j--)
            raise (SIGSTOP);
        j = interrupt_signal;
        if (j)
            raise (j);
    }

    if (dired)
    {
        /* No need to free these since we're about to exit. */
        dired_dump_obstack ("//DIRED//", &dired_obstack);
        dired_dump_obstack ("//SUBDIRED//", &subdired_obstack);
        printf ("//DIRED-OPTIONS// --quoting-style=%s\n",
                quoting_style_args[get_quoting_style (filename_quoting_options)]);
    }

    if (LOOP_DETECT)
    {
        assert (hash_get_n_entries (active_dir_set) == 0);
        hash_free (active_dir_set);
    }

    return exit_status;
}

/* Set the line length to the value given by SPEC. Return true if
   successful. 0 means no limit on line length. */

static bool
set_line_length (char const *spec)
{
    uintmax_t val;

    /* Treat too-large values as if they were SIZE_MAX, which is
       effectively infinity. */
    switch (xstrtoumax (spec, NULL, 0, &val, ""))
    {
        case LONGINT_OK:
            line_length = MIN (val, SIZE_MAX);
            return true;

        case LONGINT_OVERFLOW:
            line_length = SIZE_MAX;
            return true;

        default:
            return false;
    }
}

/* Set all the option flags according to the switches specified.
   Return the index of the first non-option argument. */

static int

```

```

decode_switches (int argc, char **argv)
{
    char *time_style_option = NULL;

    bool sort_type_specified = false;
    bool kibibytes_specified = false;

    qmark_funny_chars = false;

    /* initialize all switches to default settings */

    switch (ls_mode)
    {
        case LS_MULTI_COL:
            /* This is for the 'dir' program. */
            format = many_per_line;
            set_quoting_style (NULL, escape_quoting_style);
            break;

        case LS_LONG_FORMAT:
            /* This is for the 'vdir' program. */
            format = long_format;
            set_quoting_style (NULL, escape_quoting_style);
            break;

        case LS_LS:
            /* This is for the 'ls' program. */
            if (isatty (STDOUT_FILENO))
            {
                format = many_per_line;
                /* See description of qmark_funny_chars, above. */
                qmark_funny_chars = true;
            }
            else
            {
                format = one_per_line;
                qmark_funny_chars = false;
            }
            break;

        default:
            abort ();
    }

    time_type = time_mtime;
    sort_type = sort_name;
    sort_reverse = false;
    numeric_ids = false;
    print_block_size = false;
    indicator_style = none;
    print_inode = false;
    dereference = DEREf_UNDEFINED;
    recursive = false;
    immediate_dirs = false;
    ignore_mode = IGNORE_DEFAULT;

```

```

ignore_patterns = NULL;
hide_patterns = NULL;
print_scontext = false;

getenv_quoting_style ();

line_length = 80;
{
    char const *p = getenv ("COLUMNS");
    if (p && *p && ! set_line_length (p))
        error (0, 0,
            _("ignoring invalid width in environment variable COLUMNS: %s"),
            quote (p));
}

#ifdef TIOCGWINSZ
{
    struct winsize ws;

    if (ioctl (STDOUT_FILENO, TIOCGWINSZ, &ws) != -1
        && 0 < ws.ws_col && ws.ws_col == (size_t) ws.ws_col)
        line_length = ws.ws_col;
}
#endif

{
    char const *p = getenv ("TABSIZEL");
    tabsize = 8;
    if (p)
    {
        unsigned long int tmp_ulong;
        if (xstrtoul (p, NULL, 0, &tmp_ulong, NULL) == LONGINT_OK
            && tmp_ulong <= SIZE_MAX)
        {
            tabsize = tmp_ulong;
        }
        else
        {
            error (0, 0,
                _("ignoring invalid tab size in environment variable TABSIZE: %s"),
                quote (p));
        }
    }
}

while (true)
{
    int oi = -1;
    int c = getopt_long (argc, argv,
        "abdfghiklmnopqrstuvw:xyzABCDEFGLH:LNQRST:UXZ1",
        long_options, &oi);
    if (c == -1)
        break;
}

```

```

switch (c)
{
case 'a':
    ignore_mode = IGNORE_MINIMAL;
    break;

case 'b':
    set_quoting_style (NULL, escape_quoting_style);
    break;

case 'c':
    time_type = time_ctime;
    break;

case 'd':
    immediate_dirs = true;
    break;

case 'f':
    /* Same as enabling -a -U and disabling -l -s. */
    ignore_mode = IGNORE_MINIMAL;
    sort_type = sort_none;
    sort_type_specified = true;
    /* disable -l */
    if (format == long_format)
        format = (isatty (STDOUT_FILENO) ? many_per_line : one_per_line);
    print_block_size = false; /* disable -s */
    print_with_color = false; /* disable --color */
    break;

case FILE_TYPE_INDICATOR_OPTION: /* --file-type */
    indicator_style = file_type;
    break;

case 'g':
    format = long_format;
    print_owner = false;
    break;

case 'h':
    file_human_output_opts = human_output_opts =
        human_autoscale | human_SI | human_base_1024;
    file_output_block_size = output_block_size = 1;
    break;

case 'i':
    print_inode = true;
    break;

case 'k':
    kibibytes_specified = true;
    break;

case 'l':
    format = long_format;

```

```
        break;

    case 'm':
        format = with_commas;
        break;

    case 'n':
        numeric_ids = true;
        format = long_format;
        break;

    case 'o': /* Just like -l, but don't display group info. */
        format = long_format;
        print_group = false;
        break;

    case 'p':
        indicator_style = slash;
        break;

    case 'q':
        qmark_funny_chars = true;
        break;

    case 'r':
        sort_reverse = true;
        break;

    case 's':
        print_block_size = true;
        break;

    case 't':
        sort_type = sort_time;
        sort_type_specified = true;
        break;

    case 'u':
        time_type = time_atime;
        break;

    case 'v':
        sort_type = sort_version;
        sort_type_specified = true;
        break;

    case 'w':
        if (! set_line_length (optarg))
            error (LS_FAILURE, 0, "%s: %s", _("invalid line width"),
                  quote (optarg));
        break;

    case 'x':
        format = horizontal;
```

```
        break;

    case 'A':
        if (ignore_mode == IGNORE_DEFAULT)
            ignore_mode = IGNORE_DOT_AND_DOTDOT;
        break;

    case 'B':
        add_ignore_pattern ("*~");
        add_ignore_pattern (".*~");
        break;

    case 'C':
        format = many_per_line;
        break;

    case 'D':
        dired = true;
        break;

    case 'F':
        indicator_style = classify;
        break;

    case 'G': /* inhibit display of group info */
        print_group = false;
        break;

    case 'H':
        dereference = Deref_COMMAND_LINE_ARGUMENTS;
        break;

    case Deref_COMMAND_LINE_SYMLINK_TO_DIR_OPTION:
        dereference = Deref_COMMAND_LINE_SYMLINK_TO_DIR;
        break;

    case 'I':
        add_ignore_pattern (optarg);
        break;

    case 'L':
        dereference = Deref_ALWAYS;
        break;

    case 'N':
        set_quoting_style (NULL, literal_quoting_style);
        break;

    case 'Q':
        set_quoting_style (NULL, c_quoting_style);
        break;

    case 'R':
        recursive = true;
        break;
```

```
case 'S':
    sort_type = sort_size;
    sort_type_specified = true;
    break;

case 'T':
    tabsize = xnumtoumax (optarg, 0, 0, SIZE_MAX, "",
        _("invalid tab size"), LS_FAILURE);
    break;

case 'U':
    sort_type = sort_none;
    sort_type_specified = true;
    break;

case 'X':
    sort_type = sort_extension;
    sort_type_specified = true;
    break;

case 'l':
    /* -l has no effect after -l. */
    if (format != long_format)
        format = one_per_line;
    break;

case AUTHOR_OPTION:
    print_author = true;
    break;

case HIDE_OPTION:
    {
        struct ignore_pattern *hide = xmalloc (sizeof *hide);
        hide->pattern = optarg;
        hide->next = hide_patterns;
        hide_patterns = hide;
    }
    break;

case SORT_OPTION:
    sort_type = XARGMATCH ("--sort", optarg, sort_args, sort_types);
    sort_type_specified = true;
    break;

case GROUP_DIRECTORIES_FIRST_OPTION:
    directories_first = true;
    break;

case TIME_OPTION:
    time_type = XARGMATCH ("--time", optarg, time_args, time_types);
    break;

case FORMAT_OPTION:
```

```

format = XARGMATCH ("--format", optarg, format_args, format_types);
break;

case FULL_TIME_OPTION:
format = long_format;
time_style_option = bad_cast ("full-iso");
break;

case COLOR_OPTION:
{
int i;
if (optarg)
i = XARGMATCH ("--color", optarg, color_args, color_types);
else
/* Using --color with no argument is equivalent to using
--color=always. */
i = color_always;

print_with_color = (i == color_always
|| (i == color_if_tty
&& isatty (STDOUT_FILENO)));

if (print_with_color)
{
/* Don't use TAB characters in output. Some terminal
emulators can't handle the combination of tabs and
color codes on the same line. */
tabsize = 0;
}
break;
}

case INDICATOR_STYLE_OPTION:
indicator_style = XARGMATCH ("--indicator-style", optarg,
indicator_style_args,
indicator_style_types);

break;

case QUOTING_STYLE_OPTION:
set_quoting_style (NULL,
XARGMATCH ("--quoting-style", optarg,
quoting_style_args,
quoting_style_vals));

break;

case TIME_STYLE_OPTION:
time_style_option = optarg;
break;

case SHOW_CONTROL_CHARS_OPTION:
qmark_funny_chars = false;
break;

case BLOCK_SIZE_OPTION:
{

```



```

        enum strtol_error e = human_options (optarg, &human_output_opts,
                                              &output_block_size);
        if (e != LONGINT_OK)
            xstrtol_fatal (e, oi, 0, long_options, optarg);
        file_human_output_opts = human_output_opts;
        file_output_block_size = output_block_size;
    }
    break;

case SI_OPTION:
    file_human_output_opts = human_output_opts =
        human_autoscale | human_SI;
    file_output_block_size = output_block_size = 1;
    break;

case 'Z':
    print_scontext = true;
    break;

case_GETOPT_HELP_CHAR;

case_GETOPT_VERSION_CHAR (PROGRAM_NAME, AUTHORS);

default:
    usage (LS_FAILURE);
}
}

if (! output_block_size)
{
    char const *ls_block_size = getenv ("LS_BLOCK_SIZE");
    human_options (ls_block_size,
                  &human_output_opts, &output_block_size);
    if (ls_block_size || getenv ("BLOCK_SIZE"))
    {
        file_human_output_opts = human_output_opts;
        file_output_block_size = output_block_size;
    }
    if (kibibytes_specified)
    {
        human_output_opts = 0;
        output_block_size = 1024;
    }
}

/* Determine the max possible number of display columns. */
max_idx = line_length / MIN_COLUMN_WIDTH;
/* Account for first display column not having a separator,
   or line_lengths shorter than MIN_COLUMN_WIDTH. */
max_idx += line_length % MIN_COLUMN_WIDTH != 0;

filename_quoting_options = clone_quoting_options (NULL);
if (get_quoting_style (filename_quoting_options) == escape_quoting_style)
    set_char_quoting (filename_quoting_options, '\'', 1);

```

```

if (file_type <= indicator_style)
{
    char const *p;
    for (p = &"*=>@|"[indicator_style - file_type]; *p; p++)
        set_char_quoting (filename_quoting_options, *p, 1);
}

dirname_quoting_options = clone_quoting_options (NULL);
set_char_quoting (dirname_quoting_options, ':', 1);

/* --dired is meaningful only with --format=long (-l).
   Otherwise, ignore it.  FIXME: warn about this?
   Alternatively, make --dired imply --format=long? */
if (dired && format != long_format)
    dired = false;

/* If -c or -u is specified and not -l (or any other option that implies -l),
   and no sort-type was specified, then sort by the ctime (-c) or atime (-u).
   The behavior of ls when using either -c or -u but with neither -l nor -t
   appears to be unspecified by POSIX.  So, with GNU ls, '-u' alone means
   sort by atime (this is the one that's not specified by the POSIX spec),
   -lu means show atime and sort by name, -lut means show atime and sort
   by atime. */

if ((time_type == time_ctime || time_type == time_atime)
    && !sort_type_specified && format != long_format)
{
    sort_type = sort_time;
}

if (format == long_format)
{
    char *style = time_style_option;
    static char const posix_prefix[] = "posix-";

    if (!style)
        if (! (style = getenv ("TIME_STYLE")))
            style = bad_cast ("locale");

    while (STREQ_LEN (style, posix_prefix, sizeof posix_prefix - 1))
    {
        if (! hard_locale (LC_TIME))
            return optind;
        style += sizeof posix_prefix - 1;
    }

    if (*style == '+')
    {
        char *p0 = style + 1;
        char *p1 = strchr (p0, '\n');
        if (! p1)
            p1 = p0;
        else
        {
            if (strchr (p1 + 1, '\n'))

```

```

        error (LS_FAILURE, 0, _("invalid time style format %s"),
              quote (p0));
        *p1++ = '\0';
    }
    long_time_format[0] = p0;
    long_time_format[1] = p1;
}
else
{
    ptrdiff_t res = argmatch (style, time_style_args,
                              (char const *) time_style_types,
                              sizeof (*time_style_types));
    if (res < 0)
    {
        /* This whole block used to be a simple use of XARGMATCH.
           but that didn't print the "posix-"-prefixed variants or
           the "+"-prefixed format string option upon failure. */
        argmatch_invalid ("time style", style, res);

        /* The following is a manual expansion of argmatch_valid,
           but with the added "+" ... description and the [posix-]
           prefixes prepended. Note that this simplification works
           only because all four existing time_style_types values
           are distinct. */
        fputs (_("Valid arguments are:\n"), stderr);
        char const *const *p = time_style_args;
        while (*p)
            fprintf (stderr, " - [posix-]%%s\n", *p++);
        fputs (_(" - +FORMAT (e.g., +%H:%M) for a 'date'-style"
                " format\n"), stderr);
        usage (LS_FAILURE);
    }
    switch (res)
    {
    case full_iso_time_style:
        long_time_format[0] = long_time_format[1] =
            "%Y-%m-%d %H:%M:%S.%N %z";
        break;

    case long_iso_time_style:
        long_time_format[0] = long_time_format[1] = "%Y-%m-%d %H:%M";
        break;

    case iso_time_style:
        long_time_format[0] = "%Y-%m-%d ";
        long_time_format[1] = "%m-%d %H:%M";
        break;

    case locale_time_style:
        if (hard_locale (LC_TIME))
        {
            int i;
            for (i = 0; i < 2; i++)
                long_time_format[i] =

```

```

        dcgettext (NULL, long_time_format[i], LC_TIME);
    }
}

/* Note we leave %5b etc. alone so user widths/flags are honored. */
if (strstr (long_time_format[0], "%b")
    || strstr (long_time_format[1], "%b"))
    if (!abmon_init ())
        error (0, 0, _("error initializing month strings"));
}

return optind;
}

```

/* Parse a string as part of the LS_COLORS variable; this may involve decoding all kinds of escape characters. If equals_end is set an unescaped equal sign ends the string, otherwise only a : or \0 does. Set *OUTPUT_COUNT to the number of bytes output. Return true if successful.

The resulting string is *not* null-terminated, but may contain embedded nulls.

Note that both dest and src are char **; on return they point to the first free byte after the array and the character that ended the input string, respectively. */

```

static bool
get_funky_string (char **dest, const char **src, bool equals_end,
                  size_t *output_count)
{
    char num; /* For numerical codes */
    size_t count; /* Something to count with */
    enum {
        ST_GND, ST_BACKSLASH, ST_OCTAL, ST_HEX, ST_CARET, ST_END, ST_ERROR
    } state;
    const char *p;
    char *q;

    p = *src; /* We don't want to double-indirect */
    q = *dest; /* the whole darn time. */

    count = 0; /* No characters counted in yet. */
    num = 0;

    state = ST_GND; /* Start in ground state. */
    while (state < ST_END)
    {
        switch (state)
        {
            case ST_GND: /* Ground state (no escapes) */
                switch (*p)
                {
                    case '!':

```

```
case '\0':
    state = ST_END; /* End of string */
    break;
case '\\':
    state = ST_BACKSLASH; /* Backslash escape sequence */
    ++p;
    break;
case '^':
    state = ST_CARET; /* Caret escape */
    ++p;
    break;
case '=':
    if (equals_end)
    {
        state = ST_END; /* End */
        break;
    }
    /* else fall through */
default:
    *(q++) = *(p++);
    ++count;
    break;
}
break;

case ST_BACKSLASH: /* Backslash escaped character */
switch (*p)
{
    case '0':
    case '1':
    case '2':
    case '3':
    case '4':
    case '5':
    case '6':
    case '7':
        state = ST_OCTAL; /* Octal sequence */
        num = *p - '0';
        break;
    case 'x':
    case 'X':
        state = ST_HEX; /* Hex sequence */
        num = 0;
        break;
    case 'a': /* Bell */
        num = '\a';
        break;
    case 'b': /* Backspace */
        num = '\b';
        break;
    case 'e': /* Escape */
        num = 27;
        break;
    case 'f': /* Form feed */
```

```

        num = '\f';
        break;
    case 'n':/* Newline */
        num = '\n';
        break;
    case 'r':/* Carriage return */
        num = '\r';
        break;
    case 't':/* Tab */
        num = '\t';
        break;
    case 'v':/* Vtab */
        num = '\v';
        break;
    case '?':/* Delete */
        num = 127;
        break;
    case '_':/* Space */
        num = ' ';
        break;
    case '\0':/* End of string */
        state = ST_ERROR;/* Error! */
        break;
    default:/* Escaped character like \ ^ : = */
        num = *p;
        break;
    }
    if (state == ST_BACKSLASH)
    {
        *(q++) = num;
        ++count;
        state = ST_GND;
    }
    ++p;
    break;

case ST_OCTAL:/* Octal sequence */
    if (*p < '0' || *p > '7')
    {
        *(q++) = num;
        ++count;
        state = ST_GND;
    }
    else
        num = (num << 3) + (*(p++) - '0');
    break;

case ST_HEX:/* Hex sequence */
    switch (*p)
    {
        case '0':
        case '1':
        case '2':
        case '3':
        case '4':

```

```
        case '5':
        case '6':
        case '7':
        case '8':
        case '9':
            num = (num << 4) + (*(p++) - '0');
            break;
        case 'a':
        case 'b':
        case 'c':
        case 'd':
        case 'e':
        case 'f':
            num = (num << 4) + (*(p++) - 'a') + 10;
            break;
        case 'A':
        case 'B':
        case 'C':
        case 'D':
        case 'E':
        case 'F':
            num = (num << 4) + (*(p++) - 'A') + 10;
            break;
        default:
            *(q++) = num;
            ++count;
            state = ST_GND;
            break;
    }
    break;

case ST_CARET:/* Caret escape */
    state = ST_GND;/* Should be the next state... */
    if (*p >= '@' && *p <= '~')
    {
        *(q++) = *(p++) & 037;
        ++count;
    }
    else if (*p == '?')
    {
        *(q++) = 127;
        ++count;
    }
    else
        state = ST_ERROR;
    break;

default:
    abort ();
}

*dest = q;
*src = p;
```

```

    *output_count = count;

    return state != ST_ERROR;
}

enum parse_state
{
    PS_START = 1,
    PS_2,
    PS_3,
    PS_4,
    PS_DONE,
    PS_FAIL
};

/* Check if the content of TERM is a valid name in dircolors. */

static bool
known_term_type (void)
{
    char const *term = getenv ("TERM");
    if (! term || ! *term)
        return false;

    char const *line = G_line;
    while (line - G_line < sizeof (G_line))
    {
        if (STRNCMP_LIT (line, "TERM ") == 0)
        {
            if (fnmatch (line + 5, term, 0) == 0)
                return true;
        }
        line += strlen (line) + 1;
    }

    return false;
}

static void
parse_ls_color (void)
{
    const char *p; /* Pointer to character being parsed */
    char *buf; /* color_buf buffer pointer */
    int ind_no; /* Indicator number */
    char label[3]; /* Indicator label */
    struct color_ext_type *ext; /* Extension we are working on */

    if ((p = getenv ("LS_COLORS")) == NULL || *p == '\0')
    {
        /* LS_COLORS takes precedence, but if that's not set then
           honor the COLORTERM and TERM env variables so that
           we only go with the internal ANSI color codes if the
           former is non empty or the latter is set to a known value. */
        char const *colorterm = getenv ("COLORTERM");
    }

```



```

    if (! (colorterm && *colorterm) && ! known_term_type ())
        print_with_color = false;
    return;
}

ext = NULL;
strcpy (label, "??");

/* This is an overly conservative estimate, but any possible
   LS_COLORS string will *not* generate a color_buf longer than
   itself, so it is a safe way of allocating a buffer in
   advance. */
buf = color_buf = xstrdup (p);

enum parse_state state = PS_START;
while (true)
{
    switch (state)
    {
        case PS_START: /* First label character */
            switch (*p)
            {
                case '!':
                    ++p;
                    break;

                case '*':
                    /* Allocate new extension block and add to head of
                       linked list (this way a later definition will
                       override an earlier one, which can be useful for
                       having terminal-specific defs override global). */
                    ext = xmalloc (sizeof *ext);
                    ext->next = color_ext_list;
                    color_ext_list = ext;

                    ++p;
                    ext->ext.string = buf;

                    state = (get_funky_string (&buf, &p, true, &ext->ext.len)
                             ? PS_4 : PS_FAIL);
                    break;

                case '\0':
                    state = PS_DONE; /* Done! */
                    goto done;

                default: /* Assume it is file type label */
                    label[0] = *(p++);
                    state = PS_2;
                    break;
            }
            break;
    }
}
break;

```

```

case PS_2:/* Second label character */
    if (*p)
    {
        label[1] = *(p++);
        state = PS_3;
    }
    else
        state = PS_FAIL;/* Error */
    break;

case PS_3:/* Equal sign after indicator label */
    state = PS_FAIL;/* Assume failure... */
    if (*(p++) == '=')/* It *should* be... */
    {
        for (ind_no = 0; indicator_name[ind_no] != NULL; ++ind_no)
        {
            if (STREQ (label, indicator_name[ind_no]))
            {
                color_indicator[ind_no].string = buf;
                state = (get_funky_string (&buf, &p, false,
                                         &color_indicator[ind_no].len)
                        ? PS_START : PS_FAIL);
                break;
            }
        }
        if (state == PS_FAIL)
            error (0, 0, _("unrecognized prefix: %s"), quote (label));
    }
    break;

case PS_4:/* Equal sign after *.ext */
    if (*(p++) == '=')
    {
        ext->seq.string = buf;
        state = (get_funky_string (&buf, &p, false, &ext->seq.len)
                ? PS_START : PS_FAIL);
    }
    else
        state = PS_FAIL;
    break;

case PS_FAIL:
    goto done;

default:
    abort ();
}
}
done:

if (state == PS_FAIL)
{
    struct color_ext_type *e;
    struct color_ext_type *e2;

```

```

    error (0, 0,
        _("unparsable value for LS_COLORS environment variable"));
    free (color_buf);
    for (e = color_ext_list; e != NULL; /* empty */)
    {
        e2 = e;
        e = e->next;
        free (e2);
    }
    print_with_color = false;
}

if (color_indicator[C_LINK].len == 6
    && !STRNCMP_LIT (color_indicator[C_LINK].string, "target"))
    color_symlink_as_referent = true;
}

/* Set the quoting style default if the environment variable
   QUOTING_STYLE is set. */

static void
getenv_quoting_style (void)
{
    char const *q_style = getenv ("QUOTING_STYLE");
    if (q_style)
    {
        int i = ARGMATCH (q_style, quoting_style_args, quoting_style_vals);
        if (0 <= i)
            set_quoting_style (NULL, quoting_style_vals[i]);
        else
            error (0, 0,
                _("ignoring invalid value of environment variable QUOTING_STYLE: %s"),
                quote (q_style));
    }
}

/* Set the exit status to report a failure. If SERIOUS, it is a
   serious failure; otherwise, it is merely a minor problem. */

static void
set_exit_status (bool serious)
{
    if (serious)
        exit_status = LS_FAILURE;
    else if (exit_status == EXIT_SUCCESS)
        exit_status = LS_MINOR_PROBLEM;
}

/* Assuming a failure is serious if SERIOUS, use the printf-style
   MESSAGE to report the failure to access a file named FILE. Assume
   errno is set appropriately for the failure. */

static void
file_failure (bool serious, char const *message, char const *file)

```

```

{
    error (0, errno, message, quoteaf (file));
    set_exit_status (serious);
}

/* Request that the directory named NAME have its contents listed later.
   If REALNAME is nonzero, it will be used instead of NAME when the
   directory name is printed. This allows symbolic links to directories
   to be treated as regular directories but still be listed under their
   real names. NAME == NULL is used to insert a marker entry for the
   directory named in REALNAME.
   If NAME is non-NULL, we use its dev/ino information to save
   a call to stat -- when doing a recursive (-R) traversal.
   COMMAND_LINE_ARG means this directory was mentioned on the command line. */

static void
queue_directory (char const *name, char const *realname, bool command_line_arg)
{
    struct pending *new = xmalloc (sizeof *new);
    new->realname = realname ? xstrdup (realname) : NULL;
    new->name = name ? xstrdup (name) : NULL;
    new->command_line_arg = command_line_arg;
    new->next = pending_dirs;
    pending_dirs = new;
}

/* Read directory NAME, and list the files in it.
   If REALNAME is nonzero, print its name instead of NAME;
   this is used for symbolic links to directories.
   COMMAND_LINE_ARG means this directory was mentioned on the command line. */

static void
print_dir (char const *name, char const *realname, bool command_line_arg)
{
    DIR *dirp;
    struct dirent *next;
    uintmax_t total_blocks = 0;
    static bool first = true;

    errno = 0;
    dirp = opendir (name);
    if (!dirp)
    {
        file_failure (command_line_arg, _("cannot open directory %s"), name);
        return;
    }

    if (LOOP_DETECT)
    {
        struct stat dir_stat;
        int fd = dirfd (dirp);

        /* If dirfd failed, endure the overhead of using stat. */
        if ((0 <= fd
            ? fstat (fd, &dir_stat)

```

```

        : stat (name, &dir_stat)) < 0)
    {
        file_failure (command_line_arg,
            _("cannot determine device and inode of %s"), name);
        closedir (dirp);
        return;
    }

    /* If we've already visited this dev/inode pair, warn that
       we've found a loop, and do not process this directory. */
    if (visit_dir (dir_stat.st_dev, dir_stat.st_ino))
    {
        error (0, 0, _("%s: not listing already-listed directory"),
            quote_f (name));
        closedir (dirp);
        set_exit_status (true);
        return;
    }

    dev_ino_push (dir_stat.st_dev, dir_stat.st_ino);
}

if (recursive || print_dir_name)
{
    if (!first)
        Dired_putchar ('\n');
    first = false;
    Dired_indent ();
    push_current_dired_pos (&subdired_obstack);
    dired_pos += quote_name (stdout, realname ? realname : name,
        dirname_quoting_options, NULL);
    push_current_dired_pos (&subdired_obstack);
    Dired_fputs_literal (":\n", stdout);
}

/* Read the directory entries, and insert the subfiles into the 'cwd_file'
   table. */

clear_files ();

while (1)
{
    /* Set errno to zero so we can distinguish between a readdir failure
       and when readdir simply finds that there are no more entries. */
    errno = 0;
    next = readdir (dirp);
    if (next)
    {
        if (!file_ignored (next->d_name))
        {
            enum filetype type = unknown;

#ifdef HAVE_STRUCT_DIRENT_D_TYPE
            switch (next->d_type)

```

```

        {
        case DT_BLK: type = blockdev;break;
        case DT_CHR: type = chardev;break;
        case DT_DIR: type = directory;break;
        case DT_FIFO: type = fifo;break;
        case DT_LNK: type = symbolic_link;break;
        case DT_REG: type = normal;break;
        case DT SOCK: type = sock;break;
#ifdef DT_WHT
        case DT_WHT: type = whiteout;break;
#endif
    }
#endif

    total_blocks += gobble_file (next->d_name, type,
                                RELIABLE_D_INO (next),
                                false, name);

    /* In this narrow case, print out each name right away, so
       ls uses constant memory while processing the entries of
       this directory. Useful when there are many (millions)
       of entries in a directory. */
    if (format == one_per_line && sort_type == sort_none
        && !print_block_size && !recursive)
    {
        /* We must call sort_files in spite of
           "sort_type == sort_none" for its initialization
           of the sorted_file vector. */
        sort_files ();
        print_current_files ();
        clear_files ();
    }
}
}
else if (errno != 0)
{
    file_failure (command_line_arg, _("reading directory %s"), name);
    if (errno != EOVERFLOW)
        break;
}
else
    break;

/* When processing a very large directory, and since we've inhibited
   interrupts, this loop would take so long that ls would be annoyingly
   uninterruptible. This ensures that it handles signals promptly. */
process_signals ();
}

if (closedir (dirp) != 0)
{
    file_failure (command_line_arg, _("closing directory %s"), name);
    /* Don't return; print whatever we got. */
}

/* Sort the directory contents. */

```

```

sort_files ();

/* If any member files are subdirectories, perhaps they should have their
   contents listed rather than being mentioned here as files. */

if (recursive)
    extract_dirs_from_files (name, false);

if (format == long_format || print_block_size)
{
    const char *p;
    char buf[LONGEST_HUMAN_READABLE + 1];

    DIRED_INDENT ();
    p = _("total");
    DIRED_FPUTS (p, stdout, strlen (p));
    DIRED_PUTCHAR (' ');
    p = human_readable (total_blocks, buf, human_output_opts,
                        ST_NBLOCKSIZE, output_block_size);
    DIRED_FPUTS (p, stdout, strlen (p));
    DIRED_PUTCHAR ('\n');
}

if (cwd_n_used)
    print_current_files ();
}

/* Add 'pattern' to the list of patterns for which files that match are
   not listed. */

static void
add_ignore_pattern (const char *pattern)
{
    struct ignore_pattern *ignore;

    ignore = xmalloc (sizeof *ignore);
    ignore->pattern = pattern;
    /* Add it to the head of the linked list. */
    ignore->next = ignore_patterns;
    ignore_patterns = ignore;
}

/* Return true if one of the PATTERNS matches FILE. */

static bool
patterns_match (struct ignore_pattern const *patterns, char const *file)
{
    struct ignore_pattern const *p;
    for (p = patterns; p; p = p->next)
        if (fnmatch (p->pattern, file, FNM_PERIOD) == 0)
            return true;
    return false;
}

```

```

/* Return true if FILE should be ignored. */

static bool
file_ignored (char const *name)
{
    return ((ignore_mode != IGNORE_MINIMAL
            && name[0] == '.'
            && (ignore_mode == IGNORE_DEFAULT || ! name[1 + (name[1] == '.')]))
        || (ignore_mode == IGNORE_DEFAULT
            && patterns_match (hide_patterns, name))
        || patterns_match (ignore_patterns, name));
}

/* POSIX requires that a file size be printed without a sign, even
   when negative. Assume the typical case where negative sizes are
   actually positive values that have wrapped around. */

static uintmax_t
unsigned_file_size (off_t size)
{
    return size + (size < 0) * ((uintmax_t) OFF_T_MAX - OFF_T_MIN + 1);
}

#ifdef HAVE_CAP
/* Return true if NAME has a capability (see linux/capability.h) */
static bool
has_capability (char const *name)
{
    char *result;
    bool has_cap;

    cap_t cap_d = cap_get_file (name);
    if (cap_d == NULL)
        return false;

    result = cap_to_text (cap_d, NULL);
    cap_free (cap_d);
    if (!result)
        return false;

    /* check if human-readable capability string is empty */
    has_cap = !!*result;

    cap_free (result);
    return has_cap;
}
#else
static bool
has_capability (char const *name _GL_UNUSED)
{
    errno = ENOTSUP;
    return false;
}
#endif

```



```

/* Enter and remove entries in the table 'cwd_file'. */

static void
free_ent (struct fileinfo *f)
{
    free (f->name);
    free (f->linkname);
    if (f->scontext != UNKNOWN_SECURITY_CONTEXT)
    {
        if (is_smack_enabled ())
            free (f->scontext);
        else
            freecon (f->scontext);
    }
}

/* Empty the table of files. */
static void
clear_files (void)
{
    size_t i;

    for (i = 0; i < cwd_n_used; i++)
    {
        struct fileinfo *f = sorted_file[i];
        free_ent (f);
    }

    cwd_n_used = 0;
    any_has_acl = false;
    inode_number_width = 0;
    block_size_width = 0;
    nlink_width = 0;
    owner_width = 0;
    group_width = 0;
    author_width = 0;
    scontext_width = 0;
    major_device_number_width = 0;
    minor_device_number_width = 0;
    file_size_width = 0;
}

/* Return true if ERR implies lack-of-support failure by a
   getxattr-calling function like getfilecon or file_has_acl. */
static bool
errno_unsupported (int err)
{
    return (err == EINVAL || err == ENOSYS || is_ENOTSUP (err));
}

/* Cache *getfilecon failure, when it's trivial to do so.
   Like getfilecon/lgetfilecon, but when F's st_dev says it's doesn't
   support getting the security context, fail with ENOTSUP immediately. */
static int

```

```

getfilecon_cache (char const *file, struct fileinfo *f, bool deref)
{
    /* st_dev of the most recently processed device for which we've
       found that []getfilecon fails indicating lack of support. */
    static dev_t unsupported_device;

    if (f->stat.st_dev == unsupported_device)
    {
        errno = ENOTSUP;
        return -1;
    }
    int r = 0;
#ifdef HAVE_SMACK
    if (is_smack_enabled ())
        r = smack_new_label_from_path (file, "security.SMACK64", deref,
                                         &f->scontext);
    else
#endif
        r = (deref
             ? getfilecon (file, &f->scontext)
             : lgetfilecon (file, &f->scontext));
    if (r < 0 && errno_unsupported (errno))
        unsupported_device = f->stat.st_dev;
    return r;
}

/* Cache file_has_acl failure, when it's trivial to do.
   Like file_has_acl, but when F's st_dev says it's on a file
   system lacking ACL support, return 0 with ENOTSUP immediately. */
static int
file_has_acl_cache (char const *file, struct fileinfo *f)
{
    /* st_dev of the most recently processed device for which we've
       found that file_has_acl fails indicating lack of support. */
    static dev_t unsupported_device;

    if (f->stat.st_dev == unsupported_device)
    {
        errno = ENOTSUP;
        return 0;
    }

    /* Zero errno so that we can distinguish between two 0-returning cases:
       "has-ACL-support, but only a default ACL" and "no ACL support". */
    errno = 0;
    int n = file_has_acl (file, &f->stat);
    if (n <= 0 && errno_unsupported (errno))
        unsupported_device = f->stat.st_dev;
    return n;
}

/* Cache has_capability failure, when it's trivial to do.
   Like has_capability, but when F's st_dev says it's on a file
   system lacking capability support, return 0 with ENOTSUP immediately. */
static bool

```

```

has_capability_cache (char const *file, struct fileinfo *f)
{
    /* st_dev of the most recently processed device for which we've
       found that has_capability fails indicating lack of support. */
    static dev_t unsupported_device;

    if (f->stat.st_dev == unsupported_device)
    {
        errno = ENOTSUP;
        return 0;
    }

    bool b = has_capability (file);
    if ( !b && errno_unsupported (errno))
        unsupported_device = f->stat.st_dev;
    return b;
}

/* Add a file to the current table of files.
   Verify that the file exists, and print an error message if it does not.
   Return the number of blocks that the file occupies. */
static uintmax_t
gobble_file (char const *name, enum filetype type, ino_t inode,
             bool command_line_arg, char const *dirname)
{
    uintmax_t blocks = 0;
    struct fileinfo *f;

    /* An inode value prior to gobble_file necessarily came from readdir,
       which is not used for command line arguments. */
    assert (! command_line_arg || inode == NOT_AN_INODE_NUMBER);

    if (cwd_n_used == cwd_n_alloc)
    {
        cwd_file = xnrealloc (cwd_file, cwd_n_alloc, 2 * sizeof *cwd_file);
        cwd_n_alloc *= 2;
    }

    f = &cwd_file[cwd_n_used];
    memset (f, '\0', sizeof *f);
    f->stat.st_ino = inode;
    f->filetype = type;

    if (command_line_arg
        || format_needs_stat
        /* When coloring a directory (we may know the type from
           direct.d_type), we have to stat it in order to indicate
           sticky and/or other-writable attributes. */
        || (type == directory && print_with_color
            && (is_colored (C_OTHER_WRITABLE)
                || is_colored (C_STICKY)
                || is_colored (C_STICKY_OTHER_WRITABLE))))
        /* When dereferencing symlinks, the inode and type must come from
           stat, but readdir provides the inode and type of lstat. */

```

```

|| ((print_inode || format_needs_type)
    && (type == symbolic_link || type == unknown)
    && (dereference == DEREF_ALWAYS
        || color_symlink_as_referent || check_symlink_color))
/* Command line dereferences are already taken care of by the above
   assertion that the inode number is not yet known. */
|| (print_inode && inode == NOT_AN_INODE_NUMBER)
|| (format_needs_type
    && (type == unknown || command_line_arg
        /* --indicator-style=classify (aka -F)
           requires that we stat each regular file
           to see if it's executable. */
        || (type == normal && (indicator_style == classify
            /* This is so that --color ends up
               highlighting files with these mode
               bits set even when options like -F are
               not specified. Note we do a redundant
               stat in the very unlikely case where
               C_CAP is set but not the others. */
            || (print_with_color
                && (is_colored(C_EXEC)
                    || is_colored(C_SETUID)
                    || is_colored(C_SETGID)
                    || is_colored(C_CAP)))
            )))))

{
/* Absolute name of this file. */
char *absolute_name;
bool do_deref;
int err;

if (name[0] == '/' || dirname[0] == 0)
    absolute_name = (char *) name;
else
{
    absolute_name = alloca (strlen (name) + strlen (dirname) + 2);
    attach (absolute_name, dirname, name);
}

switch (dereference)
{
case DEREF_ALWAYS:
    err = stat (absolute_name, &f->stat);
    do_deref = true;
    break;

case DEREF_COMMAND_LINE_ARGUMENTS:
case DEREF_COMMAND_LINE_SYMLINK_TO_DIR:
    if (command_line_arg)
    {
        bool need_lstat;
        err = stat (absolute_name, &f->stat);
        do_deref = true;
    }

```

```

        if (dereference == Deref_COMMAND_LINE_ARGUMENTS)
            break;

        need_lstat = (err < 0
                      ? errno == ENOENT
                      : ! S_ISDIR (f->stat.st_mode));
        if (!need_lstat)
            break;

        /* stat failed because of ENOENT, maybe indicating a dangling
        symlink. Or stat succeeded, ABSOLUTE_NAME does not refer to a
        directory, and --dereference-command-line-symlink-to-dir is
        in effect. Fall through so that we call lstat instead. */
    }

    default: /* Deref_NEVER */
        err = lstat (absolute_name, &f->stat);
        do_deref = false;
        break;
    }

    if (err != 0)
    {
        /* Failure to stat a command line argument leads to
        an exit status of 2. For other files, stat failure
        provokes an exit status of 1. */
        file_failure (command_line_arg,
                     _("cannot access %s"), absolute_name);
        if (command_line_arg)
            return 0;

        f->name = xstrdup (name);
        cwd_n_used++;

        return 0;
    }

    f->stat_ok = true;

    /* Note has_capability() adds around 30% runtime to 'ls --color' */
    if ((type == normal || S_ISREG (f->stat.st_mode))
        && print_with_color && is_colored (C_CAP))
        f->has_capability = has_capability_cache (absolute_name, f);

    if (format == long_format || print_scontext)
    {
        bool have_scontext = false;
        bool have_acl = false;
        int attr_len = getfilecon_cache (absolute_name, f, do_deref);
        err = (attr_len < 0);

        if (err == 0)
        {
            if (is_smack_enabled ())

```

```

        have_scontext = ! STREQ ("_", f->scontext);
    else
        have_scontext = ! STREQ ("unlabeled", f->scontext);
    }
else
{
    f->scontext = UNKNOWN_SECURITY_CONTEXT;

    /* When requesting security context information, don't make
       ls fail just because the file (even a command line argument)
       isn't on the right type of file system. I.e., a getfilecon
       failure isn't in the same class as a stat failure. */
    if (is_ENOTSUP (errno) || errno == ENODATA)
        err = 0;
}

if (err == 0 && format == long_format)
{
    int n = file_has_acl_cache (absolute_name, f);
    err = (n < 0);
    have_acl = (0 < n);
}

f->acl_type = (!have_scontext && !have_acl
               ? ACL_T_NONE
               : (have_scontext && !have_acl
                  ? ACL_T_LSM_CONTEXT_ONLY
                  : ACL_T_YES));
any_has_acl |= f->acl_type != ACL_T_NONE;

if (err)
    error (0, errno, "%s", quotef (absolute_name));
}

if (S_ISLNK (f->stat.st_mode)
    && (format == long_format || check_symlink_color))
{
    struct stat linkstats;

    get_link_name (absolute_name, f, command_line_arg);
    char *linkname = make_link_name (absolute_name, f->linkname);

    /* Avoid following symbolic links when possible, ie, when
       they won't be traced and when no indicator is needed. */
    if (linkname
        && (file_type <= indicator_style || check_symlink_color)
        && stat (linkname, &linkstats) == 0)
    {
        f->linkok = true;

        /* Symbolic links to directories that are mentioned on the
           command line are automatically traced if not being
           listed as files. */
        if (!command_line_arg || format == long_format
            || !S_ISDIR (linkstats.st_mode))

```

```

        {
            /* Get the linked-to file's mode for the filetype indicator
               in long listings. */
            f->linkmode = linkstats.st_mode;
        }
    }
    free (linkname);
}

if (S_ISLNK (f->stat.st_mode))
    f->filetype = symbolic_link;
else if (S_ISDIR (f->stat.st_mode))
    {
        if (command_line_arg && !immediate_dirs)
            f->filetype = arg_directory;
        else
            f->filetype = directory;
    }
else
    f->filetype = normal;

blocks = ST_NBLOCKS (f->stat);
if (format == long_format || print_block_size)
    {
        char buf[LONGEST_HUMAN_READABLE + 1];
        int len = mbswidth (human_readable (blocks, buf, human_output_opts,
                                             ST_NBLOCKSIZE, output_block_size),
                            0);
        if (block_size_width < len)
            block_size_width = len;
    }

if (format == long_format)
    {
        if (print_owner)
            {
                int len = format_user_width (f->stat.st_uid);
                if (owner_width < len)
                    owner_width = len;
            }

        if (print_group)
            {
                int len = format_group_width (f->stat.st_gid);
                if (group_width < len)
                    group_width = len;
            }

        if (print_author)
            {
                int len = format_user_width (f->stat.st_author);
                if (author_width < len)
                    author_width = len;
            }
    }

```

```

    }

    if (print_scontext)
    {
        int len = strlen (f->scontext);
        if (scontext_width < len)
            scontext_width = len;
    }

    if (format == long_format)
    {
        char b[INT_BUFSIZE_BOUND (uintmax_t)];
        int b_len = strlen (umaxtostr (f->stat.st_nlink, b));
        if (nlink_width < b_len)
            nlink_width = b_len;

        if (S_ISCHR (f->stat.st_mode) || S_ISBLK (f->stat.st_mode))
        {
            char buf[INT_BUFSIZE_BOUND (uintmax_t)];
            int len = strlen (umaxtostr (major (f->stat.st_rdev), buf));
            if (major_device_number_width < len)
                major_device_number_width = len;
            len = strlen (umaxtostr (minor (f->stat.st_rdev), buf));
            if (minor_device_number_width < len)
                minor_device_number_width = len;
            len = major_device_number_width + 2 + minor_device_number_width;
            if (file_size_width < len)
                file_size_width = len;
        }
        else
        {
            char buf[LONGEST_HUMAN_READABLE + 1];
            uintmax_t size = unsigned_file_size (f->stat.st_size);
            int len = mbswidth (human_readable (size, buf,
                                                file_human_output_opts,
                                                1, file_output_block_size),
                                0);
            if (file_size_width < len)
                file_size_width = len;
        }
    }
}

if (print_inode)
{
    char buf[INT_BUFSIZE_BOUND (uintmax_t)];
    int len = strlen (umaxtostr (f->stat.st_ino, buf));
    if (inode_number_width < len)
        inode_number_width = len;
}

f->name = xstrdup (name);
cwd_n_used++;

return blocks;

```



```

}

/* Return true if F refers to a directory. */
static bool
is_directory (const struct fileinfo *f)
{
    return f->filetype == directory || f->filetype == arg_directory;
}

/* Put the name of the file that FILENAME is a symbolic link to
   into the LINKNAME field of 'f'.  COMMAND_LINE_ARG indicates whether
   FILENAME is a command-line argument. */

static void
get_link_name (char const *filename, struct fileinfo *f, bool command_line_arg)
{
    f->linkname = areadlink_with_size (filename, f->stat.st_size);
    if (f->linkname == NULL)
        file_failure (command_line_arg, _("cannot read symbolic link %s"),
                      filename);
}

/* If LINKNAME is a relative name and NAME contains one or more
   leading directories, return LINKNAME with those directories
   prepended; otherwise, return a copy of LINKNAME.
   If LINKNAME is NULL, return NULL. */

static char *
make_link_name (char const *name, char const *linkname)
{
    if (!linkname)
        return NULL;

    if (IS_ABSOLUTE_FILE_NAME (linkname))
        return xstrdup (linkname);

    /* The link is to a relative name. Prepend any leading directory
       in 'name' to the link name. */
    size_t prefix_len = dir_len (name);
    if (prefix_len == 0)
        return xstrdup (linkname);

    char *p = xmalloc (prefix_len + 1 + strlen (linkname) + 1);

    /* PREFIX_LEN usually specifies a string not ending in slash.
       In that case, extend it by one, since the next byte *is* a slash.
       Otherwise, the prefix is "/", so leave the length unchanged. */
    if (! ISSLASH (name[prefix_len - 1]))
        ++prefix_len;

    stpcpy (stpncpy (p, name, prefix_len), linkname);
    return p;
}

```

```

/* Return true if the last component of NAME is '.' or '..'
   This is so we don't try to recurse on './././... */

static bool
basename_is_dot_or_dotdot (const char *name)
{
    char const *base = last_component (name);
    return dot_or_dotdot (base);
}

/* Remove any entries from CWD_FILE that are for directories,
   and queue them to be listed as directories instead.
   DIRNAME is the prefix to prepend to each dirname
   to make it correct relative to ls's working dir;
   if it is null, no prefix is needed and "." and ".." should not be ignored.
   If COMMAND_LINE_ARG is true, this directory was mentioned at the top level,
   This is desirable when processing directories recursively. */

static void
extract_dirs_from_files (char const *dirname, bool command_line_arg)
{
    size_t i;
    size_t j;
    bool ignore_dot_and_dot_dot = (dirname != NULL);

    if (dirname && LOOP_DETECT)
    {
        /* Insert a marker entry first. When we dequeue this marker entry,
           we'll know that DIRNAME has been processed and may be removed
           from the set of active directories. */
        queue_directory (NULL, dirname, false);
    }

    /* Queue the directories last one first, because queueing reverses the
       order. */
    for (i = cwd_n_used; i-- != 0; )
    {
        struct fileinfo *f = sorted_file[i];

        if (is_directory (f)
            && (! ignore_dot_and_dot_dot
                || ! basename_is_dot_or_dotdot (f->name)))
        {
            if (!dirname || f->name[0] == '/')
                queue_directory (f->name, f->linkname, command_line_arg);
            else
            {
                char *name = file_name_concat (dirname, f->name, NULL);
                queue_directory (name, f->linkname, command_line_arg);
                free (name);
            }
            if (f->filetype == arg_directory)
                free_ent (f);
        }
    }
}

```

```

/* Now delete the directories from the table, compacting all the remaining
   entries. */

for (i = 0, j = 0; i < cwd_n_used; i++)
{
    struct fileinfo *f = sorted_file[i];
    sorted_file[j] = f;
    j += (f->filetype != arg_directory);
}
cwd_n_used = j;
}

/* Use strcoll to compare strings in this locale. If an error occurs,
   report an error and longjmp to failed_strcoll. */

static jmp_buf failed_strcoll;

static int
xstrcoll (char const *a, char const *b)
{
    int diff;
    errno = 0;
    diff = strcoll (a, b);
    if (errno)
    {
        error (0, errno, _("cannot compare file names %s and %s"),
              quote_n (0, a), quote_n (1, b));
        set_exit_status (false);
        longjmp (failed_strcoll, 1);
    }
    return diff;
}

/* Comparison routines for sorting the files. */

typedef void const *V;
typedef int (*qsortFunc)(V a, V b);

/* Used below in DEFINE_SORT_FUNCTIONS for _df_sort function variants.
   The do { ... } while(0) makes it possible to use the macro more like
   a statement, without violating C89 rules: */
#define DIRFIRST_CHECK(a, b)\
do\
{\
    bool a_is_dir = is_directory ((struct fileinfo const *) a);\
    bool b_is_dir = is_directory ((struct fileinfo const *) b);\
    if (a_is_dir && !b_is_dir)\
        return -1; /* a goes before b */\
    if (!a_is_dir && b_is_dir)\
        return 1; /* b goes before a */\
}\
while (0)

```

```

/* Define the 8 different sort function variants required for each sortkey.
KEY_NAME is a token describing the sort key, e.g., ctime, atime, size.
KEY_CMP_FUNC is a function to compare records based on that key, e.g.,
ctime_cmp, atime_cmp, size_cmp. Append KEY_NAME to the string,
'[rev_]str{cmp|coll}[_df]_', to create each function name. */
#define DEFINE_SORT_FUNCTIONS(key_name, key_cmp_func)\
/* direct, non-dirfirst versions */\
static int xstrcoll_##key_name (V a, V b)\
{ return key_cmp_func (a, b, xstrcoll); }\
static int strcmp_##key_name (V a, V b)\
{ return key_cmp_func (a, b, strcmp); }\
\
/* reverse, non-dirfirst versions */\
static int rev_xstrcoll_##key_name (V a, V b)\
{ return key_cmp_func (b, a, xstrcoll); }\
static int rev_strcmp_##key_name (V a, V b)\
{ return key_cmp_func (b, a, strcmp); }\
\
/* direct, dirfirst versions */\
static int xstrcoll_df_##key_name (V a, V b)\
{ DIRFIRST_CHECK (a, b); return key_cmp_func (a, b, xstrcoll); }\
static int strcmp_df_##key_name (V a, V b)\
{ DIRFIRST_CHECK (a, b); return key_cmp_func (a, b, strcmp); }\
\
/* reverse, dirfirst versions */\
static int rev_xstrcoll_df_##key_name (V a, V b)\
{ DIRFIRST_CHECK (a, b); return key_cmp_func (b, a, xstrcoll); }\
static int rev_strcmp_df_##key_name (V a, V b)\
{ DIRFIRST_CHECK (a, b); return key_cmp_func (b, a, strcmp); }

static inline int
cmp_ctime (struct fileinfo const *a, struct fileinfo const *b,
int (*cmp) (char const *, char const *))
{
int diff = timespec_cmp (get_stat_ctime (&b->stat),
get_stat_ctime (&a->stat));
return diff ? diff : cmp (a->name, b->name);
}

static inline int
cmp_mtime (struct fileinfo const *a, struct fileinfo const *b,
int (*cmp) (char const *, char const *))
{
int diff = timespec_cmp (get_stat_mtime (&b->stat),
get_stat_mtime (&a->stat));
return diff ? diff : cmp (a->name, b->name);
}

static inline int
cmp_atime (struct fileinfo const *a, struct fileinfo const *b,
int (*cmp) (char const *, char const *))
{
int diff = timespec_cmp (get_stat_atime (&b->stat),
get_stat_atime (&a->stat));
return diff ? diff : cmp (a->name, b->name);
}

```

```

}

static inline int
cmp_size (struct fileinfo const *a, struct fileinfo const *b,
          int (*cmp) (char const *, char const *))
{
    int diff = longdiff (b->stat.st_size, a->stat.st_size);
    return diff ? diff : cmp (a->name, b->name);
}

static inline int
cmp_name (struct fileinfo const *a, struct fileinfo const *b,
          int (*cmp) (char const *, char const *))
{
    return cmp (a->name, b->name);
}

/* Compare file extensions. Files with no extension are 'smallest'.
   If extensions are the same, compare by file names instead. */

static inline int
cmp_extension (struct fileinfo const *a, struct fileinfo const *b,
               int (*cmp) (char const *, char const *))
{
    char const *base1 = strrchr (a->name, '.');
    char const *base2 = strrchr (b->name, '.');
    int diff = cmp (base1 ? base1 : "", base2 ? base2 : "");
    return diff ? diff : cmp (a->name, b->name);
}

DEFINE_SORT_FUNCTIONS (ctime, cmp_ctime)
DEFINE_SORT_FUNCTIONS (mtime, cmp_mtime)
DEFINE_SORT_FUNCTIONS (atime, cmp_atime)
DEFINE_SORT_FUNCTIONS (size, cmp_size)
DEFINE_SORT_FUNCTIONS (name, cmp_name)
DEFINE_SORT_FUNCTIONS (extension, cmp_extension)

/* Compare file versions.
   Unlike all other compare functions above, cmp_version depends only
   on filevercmp, which does not fail (even for locale reasons), and does not
   need a secondary sort key. See lib/filevercmp.h for function description.

   All the other sort options, in fact, need xstrcoll and strcmp variants,
   because they all use a string comparison (either as the primary or secondary
   sort key), and xstrcoll has the ability to do a longjmp if strcoll fails for
   locale reasons. Lastly, filevercmp is ALWAYS available with gnuilib. */
static inline int
cmp_version (struct fileinfo const *a, struct fileinfo const *b)
{
    return filevercmp (a->name, b->name);
}

static int xstrcoll_version (V a, V b)
{ return cmp_version (a, b); }

```

```

static int rev_xstrcoll_version (V a, V b)
{ return cmp_version (b, a); }
static int xstrcoll_df_version (V a, V b)
{ DIRFIRST_CHECK (a, b); return cmp_version (a, b); }
static int rev_xstrcoll_df_version (V a, V b)
{ DIRFIRST_CHECK (a, b); return cmp_version (b, a); }

```

/* We have 2^3 different variants for each sort-key function
(for 3 independent sort modes).
The function pointers stored in this array must be dereferenced as:

```
sort_variants[sort_key][use_strcmp][reverse][dirs_first]
```

Note that the order in which sort keys are listed in the function pointer array below is defined by the order of the elements in the time_type and sort_type enums! */

```

#define LIST_SORTFUNCTION_VARIANTS(key_name) \
{ \
    { \
        { xstrcoll_##key_name, xstrcoll_df_##key_name }, \
        { rev_xstrcoll_##key_name, rev_xstrcoll_df_##key_name }, \
    }, \
    { \
        { strcmp_##key_name, strcmp_df_##key_name }, \
        { rev_strcmp_##key_name, rev_strcmp_df_##key_name }, \
    } \
}

```

```

static qsortFunc const sort_functions[][2][2][2] =
{
    LIST_SORTFUNCTION_VARIANTS (name),
    LIST_SORTFUNCTION_VARIANTS (extension),
    LIST_SORTFUNCTION_VARIANTS (size),

    {
        {
            { xstrcoll_version, xstrcoll_df_version },
            { rev_xstrcoll_version, rev_xstrcoll_df_version },
        },
    },

```

/* We use NULL for the strcmp variants of version comparison
since as explained in cmp_version definition, version comparison
does not rely on xstrcoll, so it will never longjmp, and never
need to try the strcmp fallback. */

```

{
    { NULL, NULL },
    { NULL, NULL },
}

```

```

/* last are time sort functions */
LIST_SORTFUNCTION_VARIANTS (mtime),
LIST_SORTFUNCTION_VARIANTS (ctime),

```

```

    LIST_SORTFUNCTION_VARIANTS (atime)
};

/* The number of sort keys is calculated as the sum of
   the number of elements in the sort_type enum (i.e., sort_numtypes)
   the number of elements in the time_type enum (i.e., time_numtypes) - 1
   This is because when sort_type==sort_time, we have up to
   time_numtypes possible sort keys.

   This line verifies at compile-time that the array of sort functions has been
   initialized for all possible sort keys. */
verify (ARRAY_CARDINALITY (sort_functions)
        == sort_numtypes + time_numtypes - 1 );

/* Set up SORTED_FILE to point to the in-use entries in CWD_FILE, in order. */

static void
initialize_ordering_vector (void)
{
    size_t i;
    for (i = 0; i < cwd_n_used; i++)
        sorted_file[i] = &cwd_file[i];
}

/* Sort the files now in the table. */

static void
sort_files (void)
{
    bool use_strcmp;

    if (sorted_file_alloc < cwd_n_used + cwd_n_used / 2)
    {
        free (sorted_file);
        sorted_file = xnmalloc (cwd_n_used, 3 * sizeof *sorted_file);
        sorted_file_alloc = 3 * cwd_n_used;
    }

    initialize_ordering_vector ();

    if (sort_type == sort_none)
        return;

    /* Try strcoll. If it fails, fall back on strcmp. We can't safely
       ignore strcoll failures, as a failing strcoll might be a
       comparison function that is not a total order, and if we ignored
       the failure this might cause qsort to dump core. */

    if (! setjmp (failed_strcoll))
        use_strcmp = false;    /* strcoll() succeeded */
    else
    {
        use_strcmp = true;
        assert (sort_type != sort_version);
    }

```

```

        initialize_ordering_vector ();
    }

    /* When sort_type == sort_time, use time_type as subindex. */
    mpsort ((void const **) sorted_file, cwd_n_used,
            sort_functions[sort_type + (sort_type == sort_time ? time_type : 0)]
            [use_strcmp][sort_reverse]
            [directories_first]);
}

/* List all the files now in the table. */

static void
print_current_files (void)
{
    size_t i;

    switch (format)
    {
        case one_per_line:
            for (i = 0; i < cwd_n_used; i++)
            {
                print_file_name_and_frills (sorted_file[i], 0);
                putchar ('\n');
            }
            break;

        case many_per_line:
            if (! line_length)
                print_with_separator (' ');
            else
                print_many_per_line ();
            break;

        case horizontal:
            if (! line_length)
                print_with_separator (' ');
            else
                print_horizontal ();
            break;

        case with_commas:
            print_with_separator (',');
            break;

        case long_format:
            for (i = 0; i < cwd_n_used; i++)
            {
                set_normal_color ();
                print_long_format (sorted_file[i]);
                DIRED_PUTCHAR ('\n');
            }
            break;
    }
}

```



```

/* Replace the first %b with precomputed aligned month names.
   Note on glibc-2.7 at least, this speeds up the whole 'ls -lU'
   process by around 17%, compared to letting strftime() handle the %b. */

static size_t
align_nstrftime (char *buf, size_t size, char const *fmt, struct tm const *tm,
                 timezone_t tz, int ns)
{
    const char *nfmt = fmt;
    /* In the unlikely event that rpl_fmt below is not large enough,
       the replacement is not done. A malloc here slows ls down by 2% */
    char rpl_fmt[sizeof (abmon[0]) + 100];
    const char *pb;
    if (required_mon_width && (pb = strstr (fmt, "%b"))
        && 0 <= tm->tm_mon && tm->tm_mon <= 11)
    {
        if (strlen (fmt) < (sizeof (rpl_fmt) - sizeof (abmon[0]) + 2))
        {
            char *pfmt = rpl_fmt;
            nfmt = rpl_fmt;

            pfmt = memcpy (pfmt, fmt, pb - fmt);
            pfmt = strcpy (pfmt, abmon[tm->tm_mon]);
            strcpy (pfmt, pb + 2);
        }
    }
    size_t ret = nstrftime (buf, size, nfmt, tm, tz, ns);
    return ret;
}

/* Return the expected number of columns in a long-format time stamp,
   or zero if it cannot be calculated. */

static int
long_time_expected_width (void)
{
    static int width = -1;

    if (width < 0)
    {
        time_t epoch = 0;
        struct tm const *tm = localtime (&epoch);
        char buf[TIME_STAMP_LEN_MAXIMUM + 1];

        /* In case you're wondering if localtime can fail with an input time_t
           value of 0, let's just say it's very unlikely, but not inconceivable.
           The TZ environment variable would have to specify a time zone that
           is 2**31-1900 years or more ahead of UTC. This could happen only on
           a 64-bit system that blindly accepts e.g., TZ=UTC+20000000000000.
           However, this is not possible with Solaris 10 or glibc-2.3.5, since
           their implementations limit the offset to 167:59 and 24:00, resp. */
        if (tm)
        {

```

```

        size_t len =
            align_nstrftime (buf, sizeof buf, long_time_format[0], tm,
                            localtime, 0);
        if (len != 0)
            width = mbsnwidth (buf, len, 0);
    }

    if (width < 0)
        width = 0;
    }

    return width;
}

/* Print the user or group name NAME, with numeric id ID, using a
   print width of WIDTH columns. */

static void
format_user_or_group (char const *name, unsigned long int id, int width)
{
    size_t len;

    if (name)
    {
        int width_gap = width - mbswidth (name, 0);
        int pad = MAX (0, width_gap);
        fputs (name, stdout);
        len = strlen (name) + pad;

        do
            putchar (' ');
        while (pad--);
    }
    else
    {
        printf ("%*lu ", width, id);
        len = width;
    }

    dired_pos += len + 1;
}

/* Print the name or id of the user with id U, using a print width of
   WIDTH. */

static void
format_user (uid_t u, int width, bool stat_ok)
{
    format_user_or_group (! stat_ok ? "?" :
                          (numeric_ids ? NULL : getuser (u)), u, width);
}

/* Likewise, for groups. */

static void

```

```

format_group (gid_t g, int width, bool stat_ok)
{
    format_user_or_group (! stat_ok ? "?" :
        (numeric_ids ? NULL : getgroup (g)), g, width);
}

/* Return the number of columns that format_user_or_group will print. */

static int
format_user_or_group_width (char const *name, unsigned long int id)
{
    if (name)
    {
        int len = mbswidth (name, 0);
        return MAX (0, len);
    }
    else
    {
        char buf[INT_BUFSIZE_BOUND (id)];
        sprintf (buf, "%lu", id);
        return strlen (buf);
    }
}

/* Return the number of columns that format_user will print. */

static int
format_user_width (uid_t u)
{
    return format_user_or_group_width (numeric_ids ? NULL : getuser (u), u);
}

/* Likewise, for groups. */

static int
format_group_width (gid_t g)
{
    return format_user_or_group_width (numeric_ids ? NULL : getgroup (g), g);
}

/* Return a pointer to a formatted version of F->stat.st_ino,
   possibly using buffer, BUF, of length BUFLen, which must be at least
   INT_BUFSIZE_BOUND (uintmax_t) bytes. */
static char *
format_inode (char *buf, size_t buflen, const struct fileinfo *f)
{
    assert (INT_BUFSIZE_BOUND (uintmax_t) <= buflen);
    return (f->stat_ok && f->stat.st_ino != NOT_AN_INODE_NUMBER
        ? umaxtostr (f->stat.st_ino, buf)
        : (char *) "?");
}

/* Print information about F in long format. */
static void

```

```

print_long_format (const struct fileinfo *f)
{
    char modebuf[12];
    char buf
    [LONGEST_HUMAN_READABLE + 1/* inode */
    + LONGEST_HUMAN_READABLE + 1/* size in blocks */
    + sizeof(modebuf) - 1 + 1/* mode string */
    + INT_BUFSIZE_BOUND (uintmax_t)/* st_nlink */
    + LONGEST_HUMAN_READABLE + 2/* major device number */
    + LONGEST_HUMAN_READABLE + 1/* minor device number */
    + TIME_STAMP_LEN_MAXIMUM + 1/* max length of time/date */
    ];
    size_t s;
    char *p;
    struct timespec when_timespec;
    struct tm *when_local;

    /* Compute the mode string, except remove the trailing space if no
       file in this directory has an ACL or security context. */
    if (f->stat_ok)
        filemodestring (&f->stat, modebuf);
    else
    {
        modebuf[0] = filetype_letter[f->filetype];
        memset (modebuf + 1, '?', 10);
        modebuf[11] = '\0';
    }
    if (! any_has_acl)
        modebuf[10] = '\0';
    else if (f->acl_type == ACL_T_LSM_CONTEXT_ONLY)
        modebuf[10] = '.';
    else if (f->acl_type == ACL_T_YES)
        modebuf[10] = '+';

    switch (time_type)
    {
        case time_ctime:
            when_timespec = get_stat_ctime (&f->stat);
            break;
        case time_mtime:
            when_timespec = get_stat_mtime (&f->stat);
            break;
        case time_atime:
            when_timespec = get_stat_atime (&f->stat);
            break;
        default:
            abort ();
    }

    p = buf;

    if (print_inode)
    {
        char hbuf[INT_BUFSIZE_BOUND (uintmax_t)];
        sprintf (p, "%*s ", inode_number_width,

```

```

        format_inode (hbuf, sizeof hbuf, f));
/* Increment by strlen (p) here, rather than by inode_number_width + 1.
   The latter is wrong when inode_number_width is zero. */
p += strlen (p);
}

if (print_block_size)
{
    char hbuf[LONGEST_HUMAN_READABLE + 1];
    char const *blocks =
        (! f->stat_ok
         ? "?"
         : human_readable (ST_NBLOCKS (f->stat), hbuf, human_output_opts,
                           ST_NBLOCKSIZE, output_block_size));
    int pad;
    for (pad = block_size_width - mbswidth (blocks, 0); 0 < pad; pad--)
        *p++ = ' ';
    while ((*p++ = *blocks++))
        continue;
    p[-1] = ' ';
}

/* The last byte of the mode string is the POSIX
   "optional alternate access method flag". */
{
    char hbuf[INT_BUFSIZE_BOUND (uintmax_t)];
    sprintf (p, "%s %s ", modebuf, nlink_width,
            ! f->stat_ok ? "?" : umaxtostr (f->stat.st_nlink, hbuf));
}
/* Increment by strlen (p) here, rather than by, e.g.,
   sizeof modebuf - 2 + any_has_acl + 1 + nlink_width + 1.
   The latter is wrong when nlink_width is zero. */
p += strlen (p);

DIRENTRY_INDEXTYPE ();

if (print_owner || print_group || print_author || print_scontext)
{
    DIRENTRY_FPUTS (buf, stdout, p - buf);

    if (print_owner)
        format_user (f->stat.st_uid, owner_width, f->stat_ok);

    if (print_group)
        format_group (f->stat.st_gid, group_width, f->stat_ok);

    if (print_author)
        format_user (f->stat.st_author, author_width, f->stat_ok);

    if (print_scontext)
        format_user_or_group (f->scontext, 0, scontext_width);

    p = buf;
}

```

```

if (f->stat_ok
    && (S_ISCHR (f->stat.st_mode) || S_ISBLK (f->stat.st_mode)))
{
    char majorbuf[INT_BUFSIZE_BOUND (uintmax_t)];
    char minorbuf[INT_BUFSIZE_BOUND (uintmax_t)];
    int blanks_width = (file_size_width
        - (major_device_number_width + 2
            + minor_device_number_width));
    sprintf (p, "%*s, %*s ",
        major_device_number_width + MAX (0, blanks_width),
        umaxtostr (major (f->stat.st_rdev), majorbuf),
        minor_device_number_width,
        umaxtostr (minor (f->stat.st_rdev), minorbuf));
    p += file_size_width + 1;
}
else
{
    char hbuf[LONGEST_HUMAN_READABLE + 1];
    char const *size =
        (! f->stat_ok
         ? "?"
         : human_readable (unsigned_file_size (f->stat.st_size),
             hbuf, file_human_output_opts, 1,
             file_output_block_size));

    int pad;
    for (pad = file_size_width - mbswidth (size, 0); 0 < pad; pad--)
        *p++ = ' ';
    while ((*p++ = *size++))
        continue;
    p[-1] = ' ';
}

when_local = localtime (&when_timespec.tv_sec);
s = 0;
*p = '\1';

if (f->stat_ok && when_local)
{
    struct timespec six_months_ago;
    bool recent;
    char const *fmt;

    /* If the file appears to be in the future, update the current
       time, in case the file happens to have been modified since
       the last time we checked the clock. */
    if (timespec_cmp (current_time, when_timespec) < 0)
    {
        /* Note that gettimeofday may call gettimeofday which, on some non-
           compliant systems, clobbers the buffer used for localtime's result.
           But it's ok here, because we use a gettimeofday wrapper that
           saves and restores the buffer around the gettimeofday call. */
        gettimeofday (&current_time);
    }
}

```

```

/* Consider a time to be recent if it is within the past six months.
   A Gregorian year has 365.2425 * 24 * 60 * 60 == 31556952 seconds
   on the average. Write this value as an integer constant to
   avoid floating point hassles. */
six_months_ago.tv_sec = current_time.tv_sec - 31556952 / 2;
six_months_ago.tv_nsec = current_time.tv_nsec;

recent = (timespec_cmp (six_months_ago, when_timespec) < 0
          && (timespec_cmp (when_timespec, current_time) < 0));
fmt = long_time_format[recent];

/* We assume here that all time zones are offset from UTC by a
   whole number of seconds. */
s = align_nstrftime (p, TIME_STAMP_LEN_MAXIMUM + 1, fmt,
                    when_local, localtime, when_timespec.tv_nsec);
}

if (s || !*p)
{
    p += s;
    *p++ = ' ';

    /* NUL-terminate the string -- fputs (via Dired_Fputs) requires it. */
    *p = '\0';
}
else
{
    /* The time cannot be converted using the desired format, so
       print it as a huge integer number of seconds. */
    char hbuf[INT_BUFSIZE_BOUND (intmax_t)];
    sprintf (p, "%*s ", long_time_expected_width (),
            (! f->stat_ok
             ? "?"
             : timetostr (when_timespec.tv_sec, hbuf)));
    /* FIXME: (maybe) We discarded when_timespec.tv_nsec. */
    p += strlen (p);
}

Dired_Fputs (buf, stdout, p - buf);
size_t w = print_name_with_quoting (f, false, &dired_obstack, p - buf);

if (f->filetype == symbolic_link)
{
    if (f->linkname)
    {
        Dired_Fputs_Literal (" -> ", stdout);
        print_name_with_quoting (f, true, NULL, (p - buf) + w + 4);
        if (indicator_style != none)
            print_type_indicator (true, f->linkmode, unknown);
    }
}
else if (indicator_style != none)
    print_type_indicator (f->stat_ok, f->stat.st_mode, f->filetype);
}

```

```

/* Output to OUT a quoted representation of the file name NAME,
   using OPTIONS to control quoting. Produce no output if OUT is NULL.
   Store the number of screen columns occupied by NAME's quoted
   representation into WIDTH, if non-NULL. Return the number of bytes
   produced. */

static size_t
quote_name (FILE *out, const char *name, struct quoting_options const *options,
            size_t *width)
{
    char smallbuf[BUFSIZ];
    size_t len = quotearg_buffer (smallbuf, sizeof smallbuf, name, -1, options);
    char *buf;
    size_t displayed_width IF_LINT ( = 0);

    if (len < sizeof smallbuf)
        buf = smallbuf;
    else
    {
        buf = alloca (len + 1);
        quotearg_buffer (buf, len + 1, name, -1, options);
    }

    enum quoting_style qs = get_quoting_style (options);

    if (qmark_funny_chars
        && (qs == shell_quoting_style || qs == shell_always_quoting_style
            || qs == literal_quoting_style))
    {
        if (MB_CUR_MAX > 1)
        {
            char const *p = buf;
            char const *plimit = buf + len;
            char *q = buf;
            displayed_width = 0;

            while (p < plimit)
                switch (*p)
                {
                    case ' ': case '!': case '"': case '#': case '%':
                    case '&': case '\\': case '(': case ')': case '*':
                    case '+': case ',': case '-': case '.': case '/':
                    case '0': case '1': case '2': case '3': case '4':
                    case '5': case '6': case '7': case '8': case '9':
                    case ':': case ';': case '<': case '=': case '>':
                    case '?':
                    case 'A': case 'B': case 'C': case 'D': case 'E':
                    case 'F': case 'G': case 'H': case 'I': case 'J':
                    case 'K': case 'L': case 'M': case 'N': case 'O':
                    case 'P': case 'Q': case 'R': case 'S': case 'T':
                    case 'U': case 'V': case 'W': case 'X': case 'Y':
                    case 'Z':
                    case '[': case '\\': case ']': case '^': case '_':
                    case 'a': case 'b': case 'c': case 'd': case 'e':

```



```

case 'f': case 'g': case 'h': case 'i': case 'j':
case 'k': case 'l': case 'm': case 'n': case 'o':
case 'p': case 'q': case 'r': case 's': case 't':
case 'u': case 'v': case 'w': case 'x': case 'y':
case 'z': case '{': case '|': case '}': case '~':
/* These characters are printable ASCII characters. */
*q++ = *p++;
displayed_width += 1;
break;
default:
/* If we have a multibyte sequence, copy it until we
reach its end, replacing each non-printable multibyte
character with a single question mark. */
{
mbstate_t mbstate = { 0, };
do
{
wchar_t wc;
size_t bytes;
int w;

bytes = mbrtowc (&wc, p, plimit - p, &mbstate);

if (bytes == (size_t) -1)
{
/* An invalid multibyte sequence was
encountered. Skip one input byte, and
put a question mark. */
p++;
*q++ = '?';
displayed_width += 1;
break;
}

if (bytes == (size_t) -2)
{
/* An incomplete multibyte character
at the end. Replace it entirely with
a question mark. */
p = plimit;
*q++ = '?';
displayed_width += 1;
break;
}

if (bytes == 0)
/* A null wide character was encountered. */
bytes = 1;

w = wcwidth (wc);
if (w >= 0)
{
/* A printable multibyte character.
Keep it. */

```

```

        for (; bytes > 0; --bytes)
            *q++ = *p++;
        displayed_width += w;
    }
    else
    {
        /* An unprintable multibyte character.
           Replace it entirely with a question
           mark. */
        p += bytes;
        *q++ = '?';
        displayed_width += 1;
    }
}
while (! mbsinit (&mbstate));
}
break;
}

/* The buffer may have shrunk. */
len = q - buf;
}
else
{
    char *p = buf;
    char const *plimit = buf + len;

    while (p < plimit)
    {
        if (! isprint (to_uchar (*p)))
            *p = '?';
        p++;
    }
    displayed_width = len;
}
}
else if (width != NULL)
{
    if (MB_CUR_MAX > 1)
        displayed_width = mbsnwidth (buf, len, 0);
    else
    {
        char const *p = buf;
        char const *plimit = buf + len;

        displayed_width = 0;
        while (p < plimit)
        {
            if (isprint (to_uchar (*p)))
                displayed_width++;
            p++;
        }
    }
}
}

```

```

    if (out != NULL)
        fwrite (buf, 1, len, out);
    if (width != NULL)
        *width = displayed_width;
    return len;
}

static size_t
print_name_with_quoting (const struct fileinfo *f,
                        bool symlink_target,
                        struct obstack *stack,
                        size_t start_col)
{
    const char* name = symlink_target ? f->linkname : f->name;

    bool used_color_this_time
        = (print_with_color
          && (print_color_indicator (f, symlink_target)
             || is_colored (C_NORM)));

    if (stack)
        PUSH_CURRENT_DIRENTRY_POS (stack);

    size_t width = quote_name (stdout, name, filename_quoting_options, NULL);
    dirent_pos += width;

    if (stack)
        PUSH_CURRENT_DIRENTRY_POS (stack);

    process_signals ();
    if (used_color_this_time)
    {
        prep_non_filename_text ();
        if (line_length
            && (start_col / line_length != (start_col + width - 1) / line_length))
            put_indicator (&color_indicator[C_CLR_TO_EOL]);
    }

    return width;
}

static void
prep_non_filename_text (void)
{
    if (color_indicator[C_END].string != NULL)
        put_indicator (&color_indicator[C_END]);
    else
    {
        put_indicator (&color_indicator[C_LEFT]);
        put_indicator (&color_indicator[C_RESET]);
        put_indicator (&color_indicator[C_RIGHT]);
    }
}

```

```

/* Print the file name of 'f' with appropriate quoting.
   Also print file size, inode number, and filetype indicator character,
   as requested by switches. */

static size_t
print_file_name_and_frills (const struct fileinfo *f, size_t start_col)
{
    char buf[MAX (LONGEST_HUMAN_READABLE + 1, INT_BUFSIZE_BOUND (uintmax_t))];

    set_normal_color ();

    if (print_inode)
        printf ("%*s ", format == with_commas ? 0 : inode_number_width,
                format_inode (buf, sizeof buf, f));

    if (print_block_size)
        printf ("%*s ", format == with_commas ? 0 : block_size_width,
                ! f->stat_ok ? "?"
                : human_readable (ST_NBLOCKS (f->stat), buf, human_output_opts,
                                   ST_NBLOCKSIZE, output_block_size));

    if (print_scontext)
        printf ("%*s ", format == with_commas ? 0 : scontext_width, f->scontext);

    size_t width = print_name_with_quoting (f, false, NULL, start_col);

    if (indicator_style != none)
        width += print_type_indicator (f->stat_ok, f->stat.st_mode, f->filetype);

    return width;
}

/* Given these arguments describing a file, return the single-byte
   type indicator, or 0. */
static char
get_type_indicator (bool stat_ok, mode_t mode, enum filetype type)
{
    char c;

    if (stat_ok ? S_ISREG (mode) : type == normal)
    {
        if (stat_ok && indicator_style == classify && (mode & S_IXUGO))
            c = '*';
        else
            c = 0;
    }
    else
    {
        if (stat_ok ? S_ISDIR (mode) : type == directory || type == arg_directory)
            c = '/';
        else if (indicator_style == slash)
            c = 0;
        else if (stat_ok ? S_ISLNK (mode) : type == symbolic_link)
            c = '@';
        else if (stat_ok ? S_ISFIFO (mode) : type == fifo)

```

```

        c = '|';
    else if (stat_ok ? S_ISSOCK (mode) : type == sock)
        c = '=';
    else if (stat_ok && S_ISDOOR (mode))
        c = '>';
    else
        c = 0;
    }
    return c;
}

static bool
print_type_indicator (bool stat_ok, mode_t mode, enum filetype type)
{
    char c = get_type_indicator (stat_ok, mode, type);
    if (c)
        DIRED_PUTCHAR (c);
    return !!c;
}

/* Returns whether any color sequence was printed. */
static bool
print_color_indicator (const struct fileinfo *f, bool symlink_target)
{
    enum indicator_no type;
    struct color_ext_type *ext; /* Color extension */
    size_t len; /* Length of name */

    const char* name;
    mode_t mode;
    int linkok;
    if (symlink_target)
    {
        name = f->linkname;
        mode = f->linkmode;
        linkok = f->linkok ? 0 : -1;
    }
    else
    {
        name = f->name;
        mode = FILE_OR_LINK_MODE (f);
        linkok = f->linkok;
    }

    /* Is this a nonexistent file? If so, linkok == -1. */

    if (linkok == -1 && is_colored (C_MISSING))
        type = C_MISSING;
    else if (!f->stat_ok)
    {
        static enum indicator_no filetype_indicator[] = FILETYPE_INDICATORS;
        type = filetype_indicator[f->filetype];
    }
    else

```

```

{
    if (S_ISREG (mode))
    {
        type = C_FILE;

        if ((mode & S_ISUID) != 0 && is_colored (C_SETUID))
            type = C_SETUID;
        else if ((mode & S_ISGID) != 0 && is_colored (C_SETGID))
            type = C_SETGID;
        else if (is_colored (C_CAP) && f->has_capability)
            type = C_CAP;
        else if ((mode & S_IXUGO) != 0 && is_colored (C_EXEC))
            type = C_EXEC;
        else if ((1 < f->stat.st_nlink) && is_colored (C_MULTIHARDLINK))
            type = C_MULTIHARDLINK;
    }
    else if (S_ISDIR (mode))
    {
        type = C_DIR;

        if ((mode & S_ISVTX) && (mode & S_IWOTH)
            && is_colored (C_STICKY_OTHER_WRITABLE))
            type = C_STICKY_OTHER_WRITABLE;
        else if ((mode & S_IWOTH) != 0 && is_colored (C_OTHER_WRITABLE))
            type = C_OTHER_WRITABLE;
        else if ((mode & S_ISVTX) != 0 && is_colored (C_STICKY))
            type = C_STICKY;
    }
    else if (S_ISLNK (mode))
        type = C_LINK;
    else if (S_ISFIFO (mode))
        type = C_FIFO;
    else if (S_ISSOCK (mode))
        type = C_SOCKET;
    else if (S_ISBLK (mode))
        type = C_BLK;
    else if (S_ISCHR (mode))
        type = C_CHR;
    else if (S_ISDOOR (mode))
        type = C_DOOR;
    else
    {
        /* Classify a file of some other type as C_ORPHAN. */
        type = C_ORPHAN;
    }
}

/* Check the file's suffix only if still classified as C_FILE. */
ext = NULL;
if (type == C_FILE)
{
    /* Test if NAME has a recognized suffix. */

    len = strlen (name);
    name += len; /* Pointer to final \0. */
}

```

```

    for (ext = color_ext_list; ext != NULL; ext = ext->next)
    {
        if (ext->ext.len <= len
            && STREQ_LEN (name - ext->ext.len, ext->ext.string,
                          ext->ext.len))
            break;
    }
}

/* Adjust the color for orphaned symlinks. */
if (type == C_LINK && !linkok)
{
    if (color_symlink_as_referent || is_colored (C_ORPHAN))
        type = C_ORPHAN;
}

{
    const struct bin_str *const s
        = ext ? &(ext->seq) : &color_indicator[type];
    if (s->string != NULL)
    {
        /* Need to reset so not dealing with attribute combinations */
        if (is_colored (C_NORM))
            restore_default_color ();
        put_indicator (&color_indicator[C_LEFT]);
        put_indicator (s);
        put_indicator (&color_indicator[C_RIGHT]);
        return true;
    }
    else
        return false;
}
}

/* Output a color indicator (which may contain nulls). */
static void
put_indicator (const struct bin_str *ind)
{
    if (!used_color)
    {
        used_color = true;
        prep_non_filename_text ();
    }

    fwrite (ind->string, ind->len, 1, stdout);
}

static size_t
length_of_file_name_and_frills (const struct fileinfo *f)
{
    size_t len = 0;
    size_t name_width;
    char buf[MAX (LONGEST_HUMAN_READABLE + 1, INT_BUFSIZE_BOUND (uintmax_t))];

```

```

if (print_inode)
    len += 1 + (format == with_commas
                ? strlen (umaxtostr (f->stat.st_ino, buf))
                : inode_number_width);

if (print_block_size)
    len += 1 + (format == with_commas
                ? strlen (! f->stat_ok ? "?"
                           : human_readable (ST_NBLOCKS (f->stat), buf,
                                               human_output_opts, ST_NBLOCKSIZE,
                                               output_block_size))
                : block_size_width);

if (print_scontext)
    len += 1 + (format == with_commas ? strlen (f->scontext) : scontext_width);

quote_name (NULL, f->name, filename_quoting_options, &name_width);
len += name_width;

if (indicator_style != none)
{
    char c = get_type_indicator (f->stat_ok, f->stat.st_mode, f->filetype);
    len += (c != 0);
}

return len;
}

static void
print_many_per_line (void)
{
    size_t row; /* Current row. */
    size_t cols = calculate_columns (true);
    struct column_info const *line_fmt = &column_info[cols - 1];

    /* Calculate the number of rows that will be in each column except possibly
       for a short column on the right. */
    size_t rows = cwd_n_used / cols + (cwd_n_used % cols != 0);

    for (row = 0; row < rows; row++)
    {
        size_t col = 0;
        size_t filesno = row;
        size_t pos = 0;

        /* Print the next row. */
        while (1)
        {
            struct fileinfo const *f = sorted_file[filesno];
            size_t name_length = length_of_file_name_and_frills (f);
            size_t max_name_length = line_fmt->col_arr[col++];
            print_file_name_and_frills (f, pos);

            filesno += rows;
            if (filesno >= cwd_n_used)

```



```

        break;

        indent (pos + name_length, pos + max_name_length);
        pos += max_name_length;
    }
    putchar ('\n');
}

static void
print_horizontal (void)
{
    size_t filesno;
    size_t pos = 0;
    size_t cols = calculate_columns (false);
    struct column_info const *line_fmt = &column_info[cols - 1];
    struct fileinfo const *f = sorted_file[0];
    size_t name_length = length_of_file_name_and_frills (f);
    size_t max_name_length = line_fmt->col_arr[0];

    /* Print first entry. */
    print_file_name_and_frills (f, 0);

    /* Now the rest. */
    for (filesno = 1; filesno < cwd_n_used; ++filesno)
    {
        size_t col = filesno % cols;

        if (col == 0)
        {
            putchar ('\n');
            pos = 0;
        }
        else
        {
            indent (pos + name_length, pos + max_name_length);
            pos += max_name_length;
        }

        f = sorted_file[filesno];
        print_file_name_and_frills (f, pos);

        name_length = length_of_file_name_and_frills (f);
        max_name_length = line_fmt->col_arr[col];
    }
    putchar ('\n');
}

/* Output name + SEP + ''. */

static void
print_with_separator (char sep)
{
    size_t filesno;

```

```

size_t pos = 0;

for (filesno = 0; filesno < cwd_n_used; filesno++)
{
    struct fileinfo const *f = sorted_file[filesno];
    size_t len = line_length ? length_of_file_name_and_frills (f) : 0;

    if (filesno != 0)
    {
        char separator;

        if (! line_length
            || ((pos + len + 2 < line_length)
                && (pos <= SIZE_MAX - len - 2)))
        {
            pos += 2;
            separator = ' ';
        }
        else
        {
            pos = 0;
            separator = '\n';
        }

        putchar (sep);
        putchar (separator);
    }

    print_file_name_and_frills (f, pos);
    pos += len;
}
putchar ('\n');
}

/* Assuming cursor is at position FROM, indent up to position TO.
   Use a TAB character instead of two or more spaces whenever possible. */

static void
indent (size_t from, size_t to)
{
    while (from < to)
    {
        if (tabsize != 0 && to / tabsize > (from + 1) / tabsize)
        {
            putchar ('\t');
            from += tabsize - from % tabsize;
        }
        else
        {
            putchar (' ');
            from++;
        }
    }
}

```

```

/* Put DIRNAME/NAME into DEST, handling '.' and '/' properly. */
/* FIXME: maybe remove this function someday. See about using a
   non-malloc'ing version of file_name_concat. */

```

```

static void
attach (char *dest, const char *dirname, const char *name)
{
    const char *dirnamep = dirname;

    /* Copy dirname if it is not ".". */
    if (dirname[0] != '.' || dirname[1] != 0)
    {
        while (*dirnamep)
            *dest++ = *dirnamep++;
        /* Add '/' if 'dirname' doesn't already end with it. */
        if (dirnamep > dirname && dirnamep[-1] != '/')
            *dest++ = '/';
    }
    while (*name)
        *dest++ = *name++;
    *dest = 0;
}

```

```

/* Allocate enough column info suitable for the current number of
   files and display columns, and initialize the info to represent the
   narrowest possible columns. */

```

```

static void
init_column_info (void)
{
    size_t i;
    size_t max_cols = MIN (max_idx, cwd_n_used);

    /* Currently allocated columns in column_info. */
    static size_t column_info_alloc;

    if (column_info_alloc < max_cols)
    {
        size_t new_column_info_alloc;
        size_t *p;

        if (max_cols < max_idx / 2)
        {
            /* The number of columns is far less than the display width
               allows. Grow the allocation, but only so that it's
               double the current requirements. If the display is
               extremely wide, this avoids allocating a lot of memory
               that is never needed. */
            column_info = xnrealloc (column_info, max_cols,
                                     2 * sizeof *column_info);
            new_column_info_alloc = 2 * max_cols;
        }
        else
        {

```

```

        column_info = xnrealloc (column_info, max_idx, sizeof *column_info);
        new_column_info_alloc = max_idx;
    }

    /* Allocate the new size_t objects by computing the triangle
       formula  $n * (n + 1) / 2$ , except that we don't need to
       allocate the part of the triangle that we've already
       allocated. Check for address arithmetic overflow. */
    {
        size_t column_info_growth = new_column_info_alloc - column_info_alloc;
        size_t s = column_info_alloc + 1 + new_column_info_alloc;
        size_t t = s * column_info_growth;
        if (s < new_column_info_alloc || t / column_info_growth != s)
            xalloc_die ();
        p = xnmalloc (t / 2, sizeof *p);
    }

    /* Grow the triangle by parceling out the cells just allocated. */
    for (i = column_info_alloc; i < new_column_info_alloc; i++)
    {
        column_info[i].col_arr = p;
        p += i + 1;
    }

    column_info_alloc = new_column_info_alloc;
}

for (i = 0; i < max_cols; ++i)
{
    size_t j;

    column_info[i].valid_len = true;
    column_info[i].line_len = (i + 1) * MIN_COLUMN_WIDTH;
    for (j = 0; j <= i; ++j)
        column_info[i].col_arr[j] = MIN_COLUMN_WIDTH;
}

/* Calculate the number of columns needed to represent the current set
   of files in the current display width. */

static size_t
calculate_columns (bool by_columns)
{
    size_t filesno; /* Index into cwd_file. */
    size_t cols; /* Number of files across. */

    /* Normally the maximum number of columns is determined by the
       screen width. But if few files are available this might limit it
       as well. */
    size_t max_cols = MIN (max_idx, cwd_n_used);

    init_column_info ();

    /* Compute the maximum number of possible columns. */

```

```

for (filesno = 0; filesno < cwd_n_used; ++filesno)
{
    struct fileinfo const *f = sorted_file[filesno];
    size_t name_length = length_of_file_name_and_frills (f);
    size_t i;

    for (i = 0; i < max_cols; ++i)
    {
        if (column_info[i].valid_len)
        {
            size_t idx = (by_columns
                ? filesno / ((cwd_n_used + i) / (i + 1))
                : filesno % (i + 1));
            size_t real_length = name_length + (idx == i ? 0 : 2);

            if (column_info[i].col_arr[idx] < real_length)
            {
                column_info[i].line_len += (real_length
                    - column_info[i].col_arr[idx]);
                column_info[i].col_arr[idx] = real_length;
                column_info[i].valid_len = (column_info[i].line_len
                    < line_length);
            }
        }
    }
}

/* Find maximum allowed columns. */
for (cols = max_cols; 1 < cols; --cols)
{
    if (column_info[cols - 1].valid_len)
        break;
}

return cols;
}

void
usage (int status)
{
    if (status != EXIT_SUCCESS)
        emit_try_help ();
    else
    {
        printf (_("Usage: %s [OPTION]... [FILE]...\n"), program_name);
        fputs (_("\n\nList information about the FILES (the current directory by default).\n\nSort entries alphabetically if none of -cftuvSUX nor --sort is specified.\n\n"), stdout);

        emit_mandatory_arg_note ();

        fputs (_("\n\n-a, --all          do not ignore entries starting with .\n\n"),

```

```

-A, --almost-all      do not list implied . and ..\n\
--author              with -l, print the author of each file\n\
-b, --escape          print C-style escapes for nongraphic characters\n\
"), stdout);
    fputs (_("\n\
--block-size=SIZE     scale sizes by SIZE before printing them; e.g.,\n\
                        '--block-size=M' prints sizes in units of\n\
                        1,048,576 bytes; see SIZE format below\n\
-B, --ignore-backups  do not list implied entries ending with ~\n\
-c                  with -lt: sort by, and show, ctime (time of last\n\
                        modification of file status information);\n\
                        with -l: show ctime and sort by name;\n\
                        otherwise: sort by ctime, newest first\n\
"), stdout);
    fputs (_("\n\
-C                  list entries by columns\n\
--color[=WHEN]       colorize the output; WHEN can be 'always' (default\n\
\n\
                        if omitted), 'auto', or 'never'; more info below\n\
\n\
-d, --directory      list directories themselves, not their contents\n\
-D, --dired          generate output designed for Emacs' dired mode\n\
"), stdout);
    fputs (_("\n\
-f                  do not sort, enable -aU, disable -ls --color\n\
-F, --classify       append indicator (one of */=>@|) to entries\n\
--file-type          likewise, except do not append '*'\n\
--format=WORD        across -x, commas -m, horizontal -x, long -l,\n\
                        single-column -l, verbose -l, vertical -C\n\
--full-time          like -l --time-style=full-iso\n\
"), stdout);
    fputs (_("\n\
-g                  like -l, but do not list owner\n\
"), stdout);
    fputs (_("\n\
--group-directories-first\n\
                        group directories before files;\n\
                        can be augmented with a --sort option, but any\n\
                        use of --sort=none (-U) disables grouping\n\
"), stdout);
    fputs (_("\n\
-G, --no-group       in a long listing, don't print group names\n\
-h, --human-readable with -l and/or -s, print human readable sizes\n\
                        (e.g., 1K 234M 2G)\n\
--si                likewise, but use powers of 1000 not 1024\n\
"), stdout);
    fputs (_("\n\
-H, --dereference-command-line\n\
                        follow symbolic links listed on the command line\n\
--dereference-command-line-symlink-to-dir\n\
                        follow each command line symbolic link\n\
                        that points to a directory\n\
--hide=PATTERN       do not list implied entries matching shell PATTERN\n\
\n\
                        (overridden by -a or -A)\n\

```

```

"), stdout);
    fputs(_("
    --indicator-style=WORD  append indicator with style WORD to entry names:\n\
\n\
                           none (default), slash (-p),\n\
                           file-type (--file-type), classify (-F)\n\
-i, --inode                print the index number of each file\n\
-I, --ignore=PATTERN      do not list implied entries matching shell PATTERN\n\
\n\
-k, --kibibytes            default to 1024-byte blocks for disk usage\n\
"), stdout);
    fputs(_("
-l                        use a long listing format\n\
-L, --dereference          when showing file information for a symbolic\n\
                           link, show information for the file the link\n\
                           references rather than for the link itself\n\
-m                        fill width with a comma separated list of entries\n\
\n\
"), stdout);
    fputs(_("
-n, --numeric-uid-gid     like -l, but list numeric user and group IDs\n\
-N, --literal              print raw entry names (don't treat e.g. control\n\
                           characters specially)\n\
-o                        like -l, but do not list group information\n\
-p, --indicator-style=slash\n\
                           append / indicator to directories\n\
\n\
"), stdout);
    fputs(_("
-q, --hide-control-chars  print ? instead of nongraphic characters\n\
--show-control-chars      show nongraphic characters as-is (the default,\n\
                           unless program is 'ls' and output is a terminal)\n\
\n\
-Q, --quote-name          enclose entry names in double quotes\n\
--quoting-style=WORD      use quoting style WORD for entry names:\n\
                           literal, locale, shell, shell-always,\n\
                           shell-escape, shell-escape-always, c, escape\n\
\n\
"), stdout);
    fputs(_("
-r, --reverse             reverse order while sorting\n\
-R, --recursive           list subdirectories recursively\n\
-s, --size                print the allocated size of each file, in blocks\n\
\n\
"), stdout);
    fputs(_("
-S                        sort by file size, largest first\n\
--sort=WORD               sort by WORD instead of name: none (-U), size (-S)\n\
\n\
                           time (-t), version (-v), extension (-X)\n\
--time=WORD               with -l, show time as WORD instead of default\n\
                           modification time: atime or access or use (-u);\n\
\n\
                           ctime or status (-c); also use specified time\n\
                           as sort key if --sort=time (newest first)\n\
\n\
"), stdout);
    fputs(_("

```

```

--time-style=STYLE    with -l, show times using style STYLE:\n\
                      full-iso, long-iso, iso, locale, or +FORMAT;\n\
                      FORMAT is interpreted like in 'date'; if FORMAT\
\n\
                      is FORMAT1<newline>FORMAT2, then FORMAT1 applies\
\n\
                      to non-recent files and FORMAT2 to recent files;\
\n\
                      if STYLE is prefixed with 'posix-', STYLE\n\
                      takes effect only outside the POSIX locale\n\
"), stdout);
    fputs (_("\n\
-t                sort by modification time, newest first\n\
-T, --tabsize=COLS    assume tab stops at each COLS instead of 8\n\
"), stdout);
    fputs (_("\n\
-u                with -lt: sort by, and show, access time;\n\
                    with -l: show access time and sort by name;\n\
                    otherwise: sort by access time, newest first\n\
-U                do not sort; list entries in directory order\n\
-v                natural sort of (version) numbers within text\n\
"), stdout);
    fputs (_("\n\
-w, --width=COLS    set output width to COLS. 0 means no limit\n\
-x                list entries by lines instead of by columns\n\
-X                sort alphabetically by entry extension\n\
-Z, --context        print any security context of each file\n\
-l                list one file per line. Avoid '\n' with -q or -b\n\
\n\
"), stdout);
    fputs (HELP_OPTION_DESCRIPTION, stdout);
    fputs (VERSION_OPTION_DESCRIPTION, stdout);
    emit_size_note ();
    fputs (_("\n\
\n\
Using color to distinguish file types is disabled both by default and\n\
with --color=never. With --color=auto, ls emits color codes only when\n\
standard output is connected to a terminal. The LS_COLORS environment\n\
variable can change the settings. Use the dircolors command to set it.\n\
"), stdout);
    fputs (_("\n\
\n\
Exit status:\n\
0 if OK,\n\
1 if minor problems (e.g., cannot access subdirectory),\n\
2 if serious trouble (e.g., cannot access command-line argument).\n\
"), stdout);
    emit_ancillary_info (PROGRAM_NAME);
}
exit (status);
}

```


1.17 corosync

1.17.1 Available under license

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.18 cpio

1.18.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as

part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may

otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.19 cracklib

1.19.1 Available under license

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations
below.

When we speak of free software, we are referring to freedom of use,
not price. Our General Public Licenses are designed to make sure that
you have the freedom to distribute copies of free software (and charge
for this service if you wish); that you receive source code or can get
it if you want it; that you can change the software and use pieces of
it in new free programs; and that you are informed that you can do
these things.

To protect your rights, we need to make restrictions that forbid
distributors to deny you these rights or to ask you to surrender these
rights. These restrictions translate to certain responsibilities for
you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them
with the library after making changes to the library and recompiling

it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

^L

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in

non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

^L

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an

appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of

a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

^L

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object

file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.
^L

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

^L

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.

You are not responsible for enforcing compliance by third parties with this License.

^L

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

^L

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

^L

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>

Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.20 cronie

1.20.1 Available under license

```

/*
 * Copyright (c) 2004 by Internet Systems Consortium, Inc. ("ISC")
 * Copyright (c) 1997,2000 by Internet Software Consortium, Inc.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES
 * WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
 * MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR
 * ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
 * WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
 * ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
 * OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
 */
/*
 * Copyright (c) 1988, 1993, 1994
 * The Regents of the University of California. All rights reserved.
 *
 * This code is derived from software written by Ken Arnold and
 * published in UNIX Review, Vol. 6, No. 8.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */
/*
 * Copyright (c) 1989, 1993

```

```

*The Regents of the University of California. All rights reserved.
*
* This code is derived from software contributed to Berkeley by
* Paul Vixie.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. Neither the name of the University nor the names of its contributors
* may be used to endorse or promote products derived from this software
* without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
*@(##)bitstring.h8.1 (Berkeley) 7/19/93
*/

```

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it

if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program

specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify

it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
/* Copyright 1988,1990,1993,1994 by Paul Vixie
 * All rights reserved
 */
```

```
/*
 * Copyright (c) 2004 by Internet Systems Consortium, Inc. ("ISC")
 * Copyright (c) 1997,2000 by Internet Software Consortium, Inc.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
```



```
* THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES
* WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
* MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR
* ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
* WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
* ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
* OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
*/

/*
* Modified 2010/09/12 by Colin Dean, Durham University IT Service,
* to add clustering support.
*/

#include "config.h"

#define MAIN_PROGRAM

#include <errno.h>
#include <langinfo.h>
#include <locale.h>
#include <pwd.h>
#include <signal.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <sys/time.h>

#ifdef WITH_INOTIFY
# include <sys/inotify.h>
#endif

#ifdef HAVE_SYS_FCNTL_H
# include <sys/fcntl.h>
#endif

#include "cronie_common.h"
#include "funcs.h"
#include "globals.h"
#include "pathnames.h"

#ifdef WITH_INOTIFY
int inotify_enabled;
#else
# define inotify_enabled 0
#endif

enum timejump { negative, small, medium, large };

static void usage(void) ATTRIBUTE_NORETURN,
run_reboot_jobs(cron_db *),
```

```

find_jobs(int, cron_db *, int, int, long),
set_time(int),
cron_sleep(int, cron_db *),
sigchld_handler(int),
sighup_handler(int),
sigchld_reaper(void), sigintterm_handler(int), parse_args(int c, char *v[]);

static volatile sig_atomic_t got_sighup, got_sigchld, got_sigintterm;
static int timeRunning, virtualTime, clockTime;
static long GMTOff;
static int DisableInotify;

#ifdef WITH_INOTIFY

/*
 * Note that inotify isn't safe to use with clustering, as changes made
 * to a shared filesystem on one system cannot be relied on to be notified
 * on another system, so use of inotify is disabled at runtime if run with
 * clustering enabled.
 */

# if defined ENABLE_SYSCRONTAB
# define NUM_WATCHES 3

int wd[NUM_WATCHES];
const char *watchpaths[NUM_WATCHES] = {SPOOL_DIR, SYS_CROND_DIR, SYSCRONTAB};
# else
# define NUM_WATCHES 2
int wd[NUM_WATCHES];
const char *watchpaths[NUM_WATCHES] = {SPOOL_DIR, SYS_CROND_DIR};
# endif

static void reset_watches(void) {
    size_t i;

    for (i = 0; i < sizeof (wd) / sizeof (wd[0]); ++i) {
        wd[i] = -2;
    }
}

void set_cron_unwatched(int fd) {
    size_t i;

    for (i = 0; i < sizeof (wd) / sizeof (wd[0]); ++i) {
        if (wd[i] > 0) {
            inotify_rm_watch(fd, wd[i]);
            wd[i] = -1;
        }
    }
}

void set_cron_watched(int fd) {
    pid_t pid = getpid();
    size_t i;

```

```

    if (fd < 0) {
        inotify_enabled = 0;
        return;
    }

    for (i = 0; i < sizeof (wd) / sizeof (wd[0]); ++i) {
        int w;

        w = inotify_add_watch(fd, watchpaths[i],
            IN_CREATE | IN_CLOSE_WRITE | IN_ATTRIB | IN_MODIFY | IN_MOVED_TO |
            IN_MOVED_FROM | IN_MOVE_SELF | IN_DELETE | IN_DELETE_SELF);
        if (w < 0 && errno != ENOENT) {
            if (wd[i] != -1) {
                log_it("CRON", pid, "This directory or file can't be watched",
                    watchpaths[i], errno);
                log_it("CRON", pid, "INFO", "running without inotify support",
                    0);
            }
            inotify_enabled = 0;
            set_cron_unwatched(fd);
            return;
        }
        wd[i] = w;
    }

    if (!inotify_enabled) {
        log_it("CRON", pid, "INFO", "running with inotify support", 0);
    }

    inotify_enabled = 1;
}
#endif

static void handle_signals(cron_db * database) {
    if (got_sighup) {
        got_sighup = 0;
#ifdef WITH_INOTIFY
        /* watches must be reinstated on reload */
        if (inotify_enabled && (EnableClustering != 1)) {
            set_cron_unwatched(database->ifd);
            reset_watches();
            inotify_enabled = 0;
        }
#endif
        database->mtime = (time_t) 0;
        log_close();
    }

    if (got_sigchld) {
        got_sigchld = 0;
        sigchld_reaper();
    }
}

```

```

static void usage(void) {
    const char **dflags;

    fprintf(stderr, "Usage:\n");
    fprintf(stderr, " %s [options]\n", ProgramName);
    fprintf(stderr, "\n");
    fprintf(stderr, "Options:\n");
    fprintf(stderr, " -h      print this message\n");
    fprintf(stderr, " -i      daemon runs without inotify support\n");
    fprintf(stderr, " -m <comm> off, or specify preferred client for sending mails\n");
    fprintf(stderr, " -n      run in foreground\n");
    fprintf(stderr, " -p      permit any crontab\n");
    fprintf(stderr, " -P      use PATH=\"%s\"", _PATH_DEFPATH);
    fprintf(stderr, " -c      enable clustering support\n");
    fprintf(stderr, " -s      log into syslog instead of sending mails\n");
    fprintf(stderr, " -x <flag> print debug information\n");
    fprintf(stderr, "\n");
    fprintf(stderr, "Debugging flags are: ");
    for (dflags = DebugFlagNames; *dflags; dflags++)
        fprintf(stderr, "%s%s", *dflags, dflags[1] ? " " : "\n");
    exit(ERROR_EXIT);
}

int main(int argc, char *argv[]) {
    struct sigaction sact;
    cron_db database;
    int fd;
    char *cs;
    pid_t pid = getpid();
    long oldGMTOff;
    struct timeval tv;
    struct timezone tz;
    char buf[256];

    if ((ProgramName=strrchr(argv[0], '/')) == NULL) {
        ProgramName = argv[0];
    }
    else {
        ++ProgramName;
    }

    MailCmd[0] = '\0';
    cron_default_mail_charset[0] = '\0';

    setlocale(LC_ALL, "");

    #if defined(BSD)
    setlinebuf(stdout);
    setlinebuf(stderr);
    #endif

    SyslogOutput = 0;
    NoFork = 0;
    ChangePath = 1;
    parse_args(argc, argv);

```

```

memset((char *) &sact, 0, sizeof sact);
sigemptyset(&sact.sa_mask);
sact.sa_flags = 0;
#ifdef SA_RESTART
sact.sa_flags |= SA_RESTART;
#endif
sact.sa_handler = sigchld_handler;
(void) sigaction(SIGCHLD, &sact, NULL);
sact.sa_handler = sighup_handler;
(void) sigaction(SIGHUP, &sact, NULL);
sact.sa_handler = sigintterm_handler;
(void) sigaction(SIGINT, &sact, NULL);
(void) sigaction(SIGTERM, &sact, NULL);

acquire_daemonlock(0);
set_cron_uid();
check_spool_dir();

if (ChangePath) {
if (setenv("PATH", _PATH_DEFPATH, 1) < 0) {
log_it("CRON", pid, "DEATH", "can't setenv PATH",
errno);
exit(1);
}
}

/* Get the default locale character set for the mail
 * "Content-Type: ...; charset=" header
 */
setlocale(LC_ALL, ""); /* set locale to system defaults or to
 * that specified by any LC_* env vars */
if ((cs = nl_langinfo(CODESET)) != 0L)
strncpy(cron_default_mail_charset, cs, MAX_ENVSTR-1);
else
strcpy(cron_default_mail_charset, "US-ASCII");

/* if there are no debug flags turned on, fork as a daemon should.
 */
if (DebugFlags) {
#ifdef DEBUGGING
(void) fprintf(stderr, "[%ld] cron started\n", (long) getpid());
#endif
}
else if (NoFork == 0) {
switch (fork()) {
case -1:
log_it("CRON", pid, "DEATH", "can't fork", errno);
exit(0);
break;
case 0:
/* child process */
(void) setsid();
if ((fd = open(_PATH_DEVNULL, O_RDWR, 0)) >= 0) {

```

```

(void) dup2(fd, STDIN);
(void) dup2(fd, STDOUT);
(void) dup2(fd, STDERR);
if (fd != STDERR)
(void) close(fd);
}
log_it("CRON", getpid(), "STARTUP", PACKAGE_VERSION, 0);
break;
default:
/* parent process should just die */
_exit(0);
}
}

if (!SyslogOutput && MailCmd[0] == '\0' && access("/usr/sbin/sendmail", X_OK) != 0) {
SyslogOutput=1;
log_it("CRON", pid, "INFO", "Syslog will be used instead of sendmail.", 0);
}

pid = getpid();

/* obtain a random scaling factor for RANDOM_DELAY */
if (gettimeofday(&tv, &tz) != 0)
tv.tv_usec = 0;
srandom(pid + tv.tv_usec);
RandomScale = random() / (double)RAND_MAX;
snprintf(buf, sizeof(buf), "RANDOM_DELAY will be scaled with factor %d%% if used.", (int)(RandomScale*100));
log_it("CRON", pid, "INFO", buf, 0);

acquire_daemonlock(0);
database.head = NULL;
database.tail = NULL;
database.mtime = (time_t) 0;

load_database(&database);

fd = -1;
#ifdef WITH_INOTIFY
if (DisableInotify || EnableClustering) {
log_it("CRON", getpid(), "No inotify - daemon runs with -i or -c option",
"", 0);
}
else {
reset_watches();
database.ifd = fd = inotify_init();
fcntl(fd, F_SETFD, FD_CLOEXEC);
if (fd < 0)
log_it("CRON", pid, "INFO", "Inotify init failed", errno);
set_cron_watched(fd);
}
#endif

set_time(TRUE);
run_reboot_jobs(&database);

```

```

timeRunning = virtualTime = clockTime;
oldGMTOff = GMTOff;

/*
 * Too many clocks, not enough time (Al. Einstein)
 * These clocks are in minutes since the epoch, adjusted for timezone.
 * virtualTime: is the time it *would* be if we woke up
 * promptly and nobody ever changed the clock. It is
 * monotonically increasing... unless a timejump happens.
 * At the top of the loop, all jobs for 'virtualTime' have run.
 * timeRunning: is the time we last awakened.
 * clockTime: is the time when set_time was last called.
 */
while (!got_sigintterm) {
int timeDiff;
enum timejump wakeupKind;

/* ... wait for the time (in minutes) to change ... */
do {
cron_sleep(timeRunning + 1, &database);
set_time(FALSE);
} while (!got_sigintterm && clockTime == timeRunning);
if (got_sigintterm)
break;
timeRunning = clockTime;

/*
 * Calculate how the current time differs from our virtual
 * clock. Classify the change into one of 4 cases.
 */
timeDiff = timeRunning - virtualTime;
check_orphans(&database);
#ifdef WITH_INOTIFY
if (inotify_enabled) {
check_inotify_database(&database);
}
else {
if (load_database(&database) && (EnableClustering != 1))
/* try reinstating the watches */
set_cron_watched(fd);
}
#else
load_database(&database);
#endif

/* shortcut for the most common case */
if (timeDiff == 1) {
virtualTime = timeRunning;
oldGMTOff = GMTOff;
find_jobs(virtualTime, &database, TRUE, TRUE, oldGMTOff);
}
else {
if (timeDiff > (3 * MINUTE_COUNT) || timeDiff < -(3 * MINUTE_COUNT))
wakeupKind = large;

```

```

else if (timeDiff > 5)
wakeupKind = medium;
else if (timeDiff > 0)
wakeupKind = small;
else
wakeupKind = negative;

switch (wakeupKind) {
case small:
/*
 * case 1: timeDiff is a small positive number
 * (wokeup late) run jobs for each virtual
 * minute until caught up.
 */
Debug(DSCH, ("[%ld], normal case %d minutes to go\n",
(long) pid, timeDiff));
do {
if (job_runqueue())
sleep(10);
virtualTime++;
if (virtualTime >= timeRunning)
/* always run also the other timezone jobs in the last step */
oldGMToff = GMToff;
find_jobs(virtualTime, &database, TRUE, TRUE, oldGMToff);
} while (virtualTime < timeRunning);
break;

case medium:
/*
 * case 2: timeDiff is a medium-sized positive
 * number, for example because we went to DST
 * run wildcard jobs once, then run any
 * fixed-time jobs that would otherwise be
 * skipped if we use up our minute (possible,
 * if there are a lot of jobs to run) go
 * around the loop again so that wildcard jobs
 * have a chance to run, and we do our
 * housekeeping.
 */
Debug(DSCH, ("[%ld], DST begins %d minutes to go\n",
(long) pid, timeDiff));
/* run wildcard jobs for current minute */
find_jobs(timeRunning, &database, TRUE, FALSE, GMToff);

/* run fixed-time jobs for each minute missed */
do {
if (job_runqueue())
sleep(10);
virtualTime++;
if (virtualTime >= timeRunning)
/* always run also the other timezone jobs in the last step */
oldGMToff = GMToff;
find_jobs(virtualTime, &database, FALSE, TRUE, oldGMToff);
set_time(FALSE);
} while (virtualTime < timeRunning && clockTime == timeRunning);

```



```

break;

case negative:
/*
 * case 3: timeDiff is a small or medium-sized
 * negative num, eg. because of DST ending.
 * Just run the wildcard jobs. The fixed-time
 * jobs probably have already run, and should
 * not be repeated. Virtual time does not
 * change until we are caught up.
 */
Debug(DSCH, ("[%ld], DST ends %d minutes to go\n",
(long) pid, timeDiff));
find_jobs(timeRunning, &database, TRUE, FALSE, GMTOff);
break;
default:
/*
 * other: time has changed a *lot*,
 * jump virtual time, and run everything
 */
Debug(DSCH, ("[%ld], clock jumped\n", (long) pid));
virtualTime = timeRunning;
oldGMTOff = GMTOff;
find_jobs(timeRunning, &database, TRUE, TRUE, GMTOff);
}
}

/* Jobs to be run (if any) are loaded; clear the queue. */
job_runqueue();

handle_signals(&database);
}

#ifdef WITH_INOTIFY
if (inotify_enabled && (EnableClustering != 1))
set_cron_unwatched(fd);

if (fd >= 0 && close(fd) < 0)
log_it("CRON", pid, "INFO", "Inotify close failed", errno);
#endif

log_it("CRON", pid, "INFO", "Shutting down", 0);

(void) unlink(_PATH_CRON_PID);

return 0;
}

static void run_reboot_jobs(cron_db * db) {
user *u;
entry *e;
int reboot;
pid_t pid = getpid();

```

```

/* lock exist - skip reboot jobs */
if (access(REBOOT_LOCK, F_OK) == 0) {
    log_it("CRON", pid, "INFO",
"@reboot jobs will be run at computer's startup.", 0);
    return;
}
/* lock doesn't exist - create lock, run reboot jobs */
if ((reboot = creat(REBOOT_LOCK, S_IRUSR & S_IWUSR)) < 0)
    log_it("CRON", pid, "INFO", "Can't create lock for reboot jobs.",
    errno);
else
    close(reboot);

for (u = db->head; u != NULL; u = u->next) {
    for (e = u->crontab; e != NULL; e = e->next) {
        if (e->flags & WHEN_REBOOT)
            job_add(e, u);
    }
}
(void) job_runqueue();
}

static void find_jobs(int vtime, cron_db * db, int doWild, int doNonWild, long vGMTOff) {
    char *orig_tz, *job_tz;
    struct tm *tm;
    int minute, hour, dom, month, dow;
    user *u;
    entry *e;

/* The support for the job-specific timezones is not perfect. There will
 * be jobs missed or run twice during the DST change in the job timezone.
 * It is recommended not to schedule any jobs during the hour when
 * the DST changes happen if job-specific timezones are used.
 *
 * Make 0-based values out of tm values so we can use them as indices
 */
#define maketime(tz1, tz2) do { \
    char *t = tz1; \
    if (t != NULL && *t != '\0') { \
        setenv("TZ", t, 1); \
        tm = localtime(&virtualGMTSecond); \
    } else { if ((tz2) != NULL) \
        setenv("TZ", (tz2), 1); \
    else \
        unsetenv("TZ"); \
        tm = gmtime(&virtualSecond); \
    } \
    minute = tm->tm_min - FIRST_MINUTE; \
    hour = tm->tm_hour - FIRST_HOUR; \
    dom = tm->tm_mday - FIRST_DOM; \
    month = tm->tm_mon + 1 /* 0..11 -> 1..12 */ - FIRST_MONTH; \
    dow = tm->tm_wday - FIRST_DOW; \
} while (0)

    orig_tz = getenv("TZ");

```

```

/* the dom/dow situation is odd.  * * 1,15 * Sun' will run on the
 * first and fifteenth AND every Sunday; ' * * * * Sun' will run *only*
 * on Sundays; ' * * 1,15 * *' will run *only* the 1st and 15th. this
 * is why we keep 'e->dow_star' and 'e->dom_star'. yes, it's bizarre.
 * like many bizarre things, it's the standard.
 */
for (u = db->head; u != NULL; u = u->next) {
for (e = u->crontab; e != NULL; e = e->next) {
time_t virtualSecond = (vtime - e->delay) * SECONDS_PER_MINUTE;
time_t virtualGMTSecond = virtualSecond - vGMTOff;
job_tz = env_get("CRON_TZ", e->envp);
maketime(job_tz, orig_tz);

/* here we test whether time is NOW */
if (bit_test(e->minute, minute) &&
bit_test(e->hour, hour) &&
bit_test(e->month, month) &&
(((e->flags & DOM_STAR) || (e->flags & DOW_STAR))
? (bit_test(e->dow, dow) && bit_test(e->dom, dom))
: (bit_test(e->dow, dow) || bit_test(e->dom, dom))
)
) {
if (job_tz != NULL && vGMTOff != GMTOff)
/* do not try to run the jobs from different timezones
 * during the DST switch of the default timezone.
 */
continue;

if ((doNonWild &&
!(e->flags & (MIN_STAR | HR_STAR))) ||
(doWild && (e->flags & (MIN_STAR | HR_STAR))))
job_add(e, u); /* will add job, if it isn't in queue already for NOW. */
}
}
}
if (orig_tz != NULL)
setenv("TZ", orig_tz, 1);
else
unsetenv("TZ");
}

/*
 * Set StartTime and clockTime to the current time.
 * These are used for computing what time it really is right now.
 * Note that clockTime is a unix wallclock time converted to minutes.
 */
static void set_time(int initialize) {
struct tm tm;
static int isdst;

StartTime = time(NULL);

/* We adjust the time to GMT so we can catch DST changes. */

```

```

tm = *localtime(&StartTime);
if (initialize || tm.tm_isdst != isdst) {
    isdst = tm.tm_isdst;
    GMTOff = get_gmtoff(&StartTime, &tm);
    Debug(DSCH, ("[%ld] GMTOff=%ld\n", (long) getpid(), (long) GMTOff));
}
clockTime = (StartTime + GMTOff) / (time_t) SECONDS_PER_MINUTE;
}

/*
 * Try to just hit the next minute.
 */
static void cron_sleep(int target, cron_db * db) {
    time_t t1, t2;
    int seconds_to_wait;

    t1 = time(NULL) + GMTOff;
    seconds_to_wait = (int) (target * SECONDS_PER_MINUTE - t1) + 1;
    Debug(DSCH, ("[%ld] Target time=%ld, sec-to-wait=%d\n",
        (long) getpid(), (long) target * SECONDS_PER_MINUTE,
        seconds_to_wait));

    while (seconds_to_wait > 0 && seconds_to_wait < 65) {
        sleep((unsigned int) seconds_to_wait);

        if (got_sigintterm)
            return;

        /*
         * Check to see if we were interrupted by a signal.
         * If so, service the signal(s) then continue sleeping
         * where we left off.
         */
        handle_signals(db);

        t2 = time(NULL) + GMTOff;
        seconds_to_wait -= (int) (t2 - t1);
        t1 = t2;
    }

    static void sighup_handler(int x) {
        got_sighup = 1;
    }

    static void sigchld_handler(int x) {
        got_sigchld = 1;
    }

    static void sigintterm_handler(int x) {
        got_sigintterm = 1;
    }

    static void sigchld_reaper(void) {
        WAIT_T waiter;

```

```
PID_T pid;

do {
pid = waitpid(-1, &waiter, WNOHANG);
switch (pid) {
case -1:
if (errno == EINTR)
continue;
Debug(DPROC, ("[%ld] sigchld...no children\n", (long) getpid()));
break;
case 0:
Debug(DPROC, ("[%ld] sigchld...no dead kids\n", (long) getpid()));
break;
default:
Debug(DPROC,
("[%ld] sigchld...pid #%ld died, stat=%d\n",
(long) getpid(), (long) pid, WEXITSTATUS(waiter)));
break;
}
} while (pid > 0);
}

static void parse_args(int argc, char *argv[]) {
int argch;

while (-1 != (argch = getopt(argc, argv, "hnpsiPx:m:c"))) {
switch (argch) {
case 'x':
if (!set_debug_flags(optarg))
usage();
break;
case 'n':
NoFork = 1;
break;
case 'p':
PermitAnyCrontab = 1;
break;
case 's':
SyslogOutput = 1;
break;
case 'i':
DisableInotify = 1;
break;
case 'P':
ChangePath = 0;
break;
case 'm':
strncpy(MailCmd, optarg, MAX_COMMAND);
break;
case 'c':
EnableClustering = 1;
break;
case 'h':
default:
```

```

usage();
break;
}
}
}
}
/*$NetBSD: popen.c,v 1.9 2005/03/16 02:53:55 xtraeme Exp $*/

/*
 * Copyright (c) 1988, 1993, 1994
 * The Regents of the University of California. All rights reserved.
 *
 * This code is derived from software written by Ken Arnold and
 * published in UNIX Review, Vol. 6, No. 8.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include "config.h"

#include <errno.h>
#include <pwd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/wait.h>
#include <unistd.h>

#include "funcs.h"
#include "globals.h"
#include "macros.h"

#ifdef HAVE_SYS_CDEFS_H
# include <sys/cdefs.h>
#endif

```

```

#include <signal.h>

/*
 * Special version of popen which avoids call to shell. This insures noone
 * may create a pipe to a hidden program as a side effect of a list or dir
 * command.
 */
static PID_T *pids;
static int fds;

#define MAX_ARGS 1024

FILE *cron_popen(char *program, const char *type, struct passwd *pw, char **jobenv) {
    char *cp;
    FILE *iop;
    int argc, pdes[2];
    PID_T pid;
    char *argv[MAX_ARGS];
    ssize_t out;
    char buf[PIPE_BUF];
    struct sigaction sa;
    int fd;

#ifdef __GNUC__
    (void) &iop; /* Avoid fork clobbering */
#endif

    if ((*type != 'r' && *type != 'w') || type[1])
        return (NULL);

    if (!pids) {
        if ((fds = getdtablesize()) <= 0)
            return (NULL);
        if (!(pids = (PID_T *) malloc((u_int) (fds * sizeof (PID_T))))
            return (NULL);
        memset((char *) pids, 0, fds * sizeof (PID_T));
    }
    if (pipe(pdes) < 0)
        return (NULL);

    /* break up string into pieces */
    for (argc = 0, cp = program; argc < MAX_ARGS; cp = NULL)
        if (!(argv[argc++] = strtok(cp, "\t\n")))
            break;

    iop = NULL;
    switch (pid = fork()) {
        case -1: /* error */
            (void) close(pdes[0]);
            (void) close(pdes[1]);
            goto pfree;
        /* NOTREACHED */
        case 0: /* child */
            if (*type == 'r') {

```

```

if (pdes[1] != STDOUT) {
    dup2(pdes[1], STDOUT);
    dup2(pdes[1], STDERR); /* stderr, too! */
    (void) close(pdes[1]);
}
(void) close(pdes[0]);
}
else {
    if (pdes[0] != STDIN) {
        dup2(pdes[0], STDIN);
        (void) close(pdes[0]);
    }
    (void) close(pdes[1]);
}

/* reset SIGPIPE to default for the child */
memset(&sa, 0, sizeof(sa));
sa.sa_handler = SIG_DFL;
sigaction(SIGPIPE, &sa, NULL);

/* close all unwanted open file descriptors */
for (fd = STDERR + 1; fd < fds; fd++) {
    close(fd);
}

if (cron_change_user_permanently(pw, env_get("HOME", jobenv)) != 0)
    _exit(2);

if (execvpe(argv[0], argv, jobenv) < 0) {
    int save_errno = errno;

    log_it("CRON", getpid(), "EXEC FAILED", program, save_errno);
    if (*type != 'r') {
        while (0 != (out = read(STDIN, buf, PIPE_BUF))) {
            if ((out == -1) && (errno != EINTR))
                break;
        }
    }
    _exit(1);
}

/* parent; assume fdopen can't fail... */
if (*type == 'r') {
    iop = fdopen(pdes[0], type);
    (void) close(pdes[1]);
}
else {
    iop = fdopen(pdes[1], type);
    (void) close(pdes[0]);
}
pids[fileno(iop)] = pid;

pfree:
return (iop);
}

```



```
int cron_pclose(FILE * iop) {
    int fdes;
    sigset_t oset, nset;
    WAIT_T stat_loc;
    PID_T pid;

    /*
     * pclose returns -1 if stream is not associated with a
     * `popened' command, or, if already `pclosed'.
     */
    if (pids == 0 || pids[fdes = fileno(iop)] == 0)
        return (-1);
    (void) fclose(iop);

    sigemptyset(&nset);
    sigaddset(&nset, SIGINT);
    sigaddset(&nset, SIGQUIT);
    sigaddset(&nset, SIGHUP);
    (void) sigprocmask(SIG_BLOCK, &nset, &oset);
    while ((pid = wait(&stat_loc)) != pids[fdes] && pid != -1) ;
    (void) sigprocmask(SIG_SETMASK, &oset, NULL);
    pids[fdes] = 0;
    return (pid == -1 ? -1 : WEXITSTATUS(stat_loc));
}
```

1.21 curl

1.21.1 Available under license

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2016, Daniel Stenberg, <daniel@haxx.se>, and many contributors, see the THANKS file.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

1.22 cyrus-sasl

1.22.1 Available under license

```
/* CMU libsasl
 * Tim Martin
 * Rob Earhart
 * Rob Siemborski
 */
/*
 * Copyright (c) 1998-2003 Carnegie Mellon University. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in
 * the documentation and/or other materials provided with the
 * distribution.
 *
 * 3. The name "Carnegie Mellon University" must not be used to
 * endorse or promote products derived from this software without
 * prior written permission. For permission or any other legal
 * details, please contact
 * Office of Technology Transfer
 * Carnegie Mellon University
 * 5000 Forbes Avenue
 * Pittsburgh, PA 15213-3890
 * (412) 268-4387, fax: (412) 268-7395
 * tech-transfer@andrew.cmu.edu
 *
 * 4. Redistributions of any form whatsoever must retain the following
 * acknowledgment:
 * "This product includes software developed by Computing Services
 * at Carnegie Mellon University (http://www.cmu.edu/computing/)."
```

CARNEGIE MELLON UNIVERSITY DISCLAIMS ALL WARRANTIES WITH REGARD TO
THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS, IN NO EVENT SHALL CARNEGIE MELLON UNIVERSITY BE LIABLE
FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN
AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING
OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

1.23 db

1.23.1 Available under license

```
/*_
 * $Id$
 */
```

The following is the license that applies to this copy of the Berkeley DB software. For a license to use the Berkeley DB software under conditions other than those described here, or to purchase support for this software, please contact Oracle at berkeleydb-info_us@oracle.com.

If you were looking for the license that applies to Berkeley DB XML, click here.
<http://www.oracle.com/technetwork/products/berkeleydb/downloads/xmllicense-086890.html>

If you were looking for the license that applies to Berkeley DB Java Edition, click here.
<http://www.oracle.com/technetwork/products/berkeleydb/downloads/jeoslicense-086837.html>

```

/*
 * The Oracle Berkeley DB product is licensed
 * under the GNU AFFERO GENERAL PUBLIC LICENSE:
 *
 *
 *          GNU AFFERO GENERAL PUBLIC LICENSE
 *          Version 3, 19 November 2007
 *
 * Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 * Everyone is permitted to copy and distribute verbatim copies
 * of this license document, but changing it is not allowed.
 *
 *           Preamble
 *
 *   The GNU Affero General Public License is a free, copyleft license for
 * software and other kinds of works, specifically designed to ensure
 * cooperation with the community in the case of network server software.
 *
 *   The licenses for most software and other practical works are designed
 * to take away your freedom to share and change the works. By contrast,
 * our General Public Licenses are intended to guarantee your freedom to
 * share and change all versions of a program--to make sure it remains free
 * software for all its users.
 *
 *   When we speak of free software, we are referring to freedom, not
 * price. Our General Public Licenses are designed to make sure that you
 * have the freedom to distribute copies of free software (and charge for
 * them if you wish), that you receive source code or can get it if you
 * want it, that you can change the software or use pieces of it in new
 * free programs, and that you know you can do these things.
 *
 *   Developers that use our General Public Licenses protect your rights

```

- * with two steps: (1) assert copyright on the software, and (2) offer
- * you this License which gives you legal permission to copy, distribute
- * and/or modify the software.
- *
- * A secondary benefit of defending all users' freedom is that
- * improvements made in alternate versions of the program, if they
- * receive widespread use, become available for other developers to
- * incorporate. Many developers of free software are heartened and
- * encouraged by the resulting cooperation. However, in the case of
- * software used on network servers, this result may fail to come about.
- * The GNU General Public License permits making a modified version and
- * letting the public access it on a server without ever releasing its
- * source code to the public.
- *
- * The GNU Affero General Public License is designed specifically to
- * ensure that, in such cases, the modified source code becomes available
- * to the community. It requires the operator of a network server to
- * provide the source code of the modified version running there to the
- * users of that server. Therefore, public use of a modified version, on
- * a publicly accessible server, gives the public access to the source
- * code of the modified version.
- *
- * An older license, called the Affero General Public License and
- * published by Affero, was designed to accomplish similar goals. This is
- * a different license, not a version of the Affero GPL, but Affero has
- * released a new version of the Affero GPL which permits relicensing under
- * this license.
- *
- * The precise terms and conditions for copying, distribution and
- * modification follow.

TERMS AND CONDITIONS

- * 0. Definitions.
- *
- * "This License" refers to version 3 of the GNU Affero General Public License.
- *
- * "Copyright" also means copyright-like laws that apply to other kinds of
- * works, such as semiconductor masks.
- *
- * "The Program" refers to any copyrightable work licensed under this
- * License. Each licensee is addressed as "you". "Licensees" and
- * "recipients" may be individuals or organizations.
- *
- * To "modify" a work means to copy from or adapt all or part of the work
- * in a fashion requiring copyright permission, other than the making of an
- * exact copy. The resulting work is called a "modified version" of the
- * earlier work or a work "based on" the earlier work.
- *
- * A "covered work" means either the unmodified Program or a work based
- * on the Program.
- *
- * To "propagate" a work means to do anything with it that, without
- * permission, would make you directly or secondarily liable for

- * infringement under applicable copyright law, except executing it on a
- * computer or modifying a private copy. Propagation includes copying,
- * distribution (with or without modification), making available to the
- * public, and in some countries other activities as well.
- *
- * To "convey" a work means any kind of propagation that enables other
- * parties to make or receive copies. Mere interaction with a user through
- * a computer network, with no transfer of a copy, is not conveying.
- *
- * An interactive user interface displays "Appropriate Legal Notices"
- * to the extent that it includes a convenient and prominently visible
- * feature that (1) displays an appropriate copyright notice, and (2)
- * tells the user that there is no warranty for the work (except to the
- * extent that warranties are provided), that licensees may convey the
- * work under this License, and how to view a copy of this License. If
- * the interface presents a list of user commands or options, such as a
- * menu, a prominent item in the list meets this criterion.
- *
- * 1. Source Code.
- *
- * The "source code" for a work means the preferred form of the work
- * for making modifications to it. "Object code" means any non-source
- * form of a work.
- *
- * A "Standard Interface" means an interface that either is an official
- * standard defined by a recognized standards body, or, in the case of
- * interfaces specified for a particular programming language, one that
- * is widely used among developers working in that language.
- *
- * The "System Libraries" of an executable work include anything, other
- * than the work as a whole, that (a) is included in the normal form of
- * packaging a Major Component, but which is not part of that Major
- * Component, and (b) serves only to enable use of the work with that
- * Major Component, or to implement a Standard Interface for which an
- * implementation is available to the public in source code form. A
- * "Major Component", in this context, means a major essential component
- * (kernel, window system, and so on) of the specific operating system
- * (if any) on which the executable work runs, or a compiler used to
- * produce the work, or an object code interpreter used to run it.
- *
- * The "Corresponding Source" for a work in object code form means all
- * the source code needed to generate, install, and (for an executable
- * work) run the object code and to modify the work, including scripts to
- * control those activities. However, it does not include the work's
- * System Libraries, or general-purpose tools or generally available free
- * programs which are used unmodified in performing those activities but
- * which are not part of the work. For example, Corresponding Source
- * includes interface definition files associated with source files for
- * the work, and the source code for shared libraries and dynamically
- * linked subprograms that the work is specifically designed to require,
- * such as by intimate data communication or control flow between those
- * subprograms and other parts of the work.
- *
- * The Corresponding Source need not include anything that users
- * can regenerate automatically from other parts of the Corresponding

- * Source.
- *
- * The Corresponding Source for a work in source code form is that
- * same work.
- *
- * 2. Basic Permissions.
- *
- * All rights granted under this License are granted for the term of
- * copyright on the Program, and are irrevocable provided the stated
- * conditions are met. This License explicitly affirms your unlimited
- * permission to run the unmodified Program. The output from running a
- * covered work is covered by this License only if the output, given its
- * content, constitutes a covered work. This License acknowledges your
- * rights of fair use or other equivalent, as provided by copyright law.
- *
- * You may make, run and propagate covered works that you do not
- * convey, without conditions so long as your license otherwise remains
- * in force. You may convey covered works to others for the sole purpose
- * of having them make modifications exclusively for you, or provide you
- * with facilities for running those works, provided that you comply with
- * the terms of this License in conveying all material for which you do
- * not control copyright. Those thus making or running the covered works
- * for you must do so exclusively on your behalf, under your direction
- * and control, on terms that prohibit them from making any copies of
- * your copyrighted material outside their relationship with you.
- *
- * Conveying under any other circumstances is permitted solely under
- * the conditions stated below. Sublicensing is not allowed; section 10
- * makes it unnecessary.
- *
- * 3. Protecting Users' Legal Rights From Anti-Circumvention Law.
- *
- * No covered work shall be deemed part of an effective technological
- * measure under any applicable law fulfilling obligations under article
- * 11 of the WIPO copyright treaty adopted on 20 December 1996, or
- * similar laws prohibiting or restricting circumvention of such
- * measures.
- *
- * When you convey a covered work, you waive any legal power to forbid
- * circumvention of technological measures to the extent such circumvention
- * is effected by exercising rights under this License with respect to
- * the covered work, and you disclaim any intention to limit operation or
- * modification of the work as a means of enforcing, against the work's
- * users, your or third parties' legal rights to forbid circumvention of
- * technological measures.
- *
- * 4. Conveying Verbatim Copies.
- *
- * You may convey verbatim copies of the Program's source code as you
- * receive it, in any medium, provided that you conspicuously and
- * appropriately publish on each copy an appropriate copyright notice;
- * keep intact all notices stating that this License and any
- * non-permissive terms added in accord with section 7 apply to the code;
- * keep intact all notices of the absence of any warranty; and give all

- * recipients a copy of this License along with the Program.
- *
- * You may charge any price or no price for each copy that you convey,
- * and you may offer support or warranty protection for a fee.
- *
- * 5. Conveying Modified Source Versions.
- *
- * You may convey a work based on the Program, or the modifications to
- * produce it from the Program, in the form of source code under the
- * terms of section 4, provided that you also meet all of these conditions:
- *
- * a) The work must carry prominent notices stating that you modified
- * it, and giving a relevant date.
- *
- * b) The work must carry prominent notices stating that it is
- * released under this License and any conditions added under section
- * 7. This requirement modifies the requirement in section 4 to
- * "keep intact all notices".
- *
- * c) You must license the entire work, as a whole, under this
- * License to anyone who comes into possession of a copy. This
- * License will therefore apply, along with any applicable section 7
- * additional terms, to the whole of the work, and all its parts,
- * regardless of how they are packaged. This License gives no
- * permission to license the work in any other way, but it does not
- * invalidate such permission if you have separately received it.
- *
- * d) If the work has interactive user interfaces, each must display
- * Appropriate Legal Notices; however, if the Program has interactive
- * interfaces that do not display Appropriate Legal Notices, your
- * work need not make them do so.
- *
- * A compilation of a covered work with other separate and independent
- * works, which are not by their nature extensions of the covered work,
- * and which are not combined with it such as to form a larger program,
- * in or on a volume of a storage or distribution medium, is called an
- * "aggregate" if the compilation and its resulting copyright are not
- * used to limit the access or legal rights of the compilation's users
- * beyond what the individual works permit. Inclusion of a covered work
- * in an aggregate does not cause this License to apply to the other
- * parts of the aggregate.
- *
- * 6. Conveying Non-Source Forms.
- *
- * You may convey a covered work in object code form under the terms
- * of sections 4 and 5, provided that you also convey the
- * machine-readable Corresponding Source under the terms of this License,
- * in one of these ways:
- *
- * a) Convey the object code in, or embodied in, a physical product
- * (including a physical distribution medium), accompanied by the
- * Corresponding Source fixed on a durable physical medium
- * customarily used for software interchange.
- *
- * b) Convey the object code in, or embodied in, a physical product

- * (including a physical distribution medium), accompanied by a
- * written offer, valid for at least three years and valid for as
- * long as you offer spare parts or customer support for that product
- * model, to give anyone who possesses the object code either (1) a
- * copy of the Corresponding Source for all the software in the
- * product that is covered by this License, on a durable physical
- * medium customarily used for software interchange, for a price no
- * more than your reasonable cost of physically performing this
- * conveying of source, or (2) access to copy the
- * Corresponding Source from a network server at no charge.
- *
- * c) Convey individual copies of the object code with a copy of the
- * written offer to provide the Corresponding Source. This
- * alternative is allowed only occasionally and noncommercially, and
- * only if you received the object code with such an offer, in accord
- * with subsection 6b.
- *
- * d) Convey the object code by offering access from a designated
- * place (gratis or for a charge), and offer equivalent access to the
- * Corresponding Source in the same way through the same place at no
- * further charge. You need not require recipients to copy the
- * Corresponding Source along with the object code. If the place to
- * copy the object code is a network server, the Corresponding Source
- * may be on a different server (operated by you or a third party)
- * that supports equivalent copying facilities, provided you maintain
- * clear directions next to the object code saying where to find the
- * Corresponding Source. Regardless of what server hosts the
- * Corresponding Source, you remain obligated to ensure that it is
- * available for as long as needed to satisfy these requirements.
- *
- * e) Convey the object code using peer-to-peer transmission, provided
- * you inform other peers where the object code and Corresponding
- * Source of the work are being offered to the general public at no
- * charge under subsection 6d.
- *
- * A separable portion of the object code, whose source code is excluded
- * from the Corresponding Source as a System Library, need not be
- * included in conveying the object code work.
- *
- * A "User Product" is either (1) a "consumer product", which means any
- * tangible personal property which is normally used for personal, family,
- * or household purposes, or (2) anything designed or sold for incorporation
- * into a dwelling. In determining whether a product is a consumer product,
- * doubtful cases shall be resolved in favor of coverage. For a particular
- * product received by a particular user, "normally used" refers to a
- * typical or common use of that class of product, regardless of the status
- * of the particular user or of the way in which the particular user
- * actually uses, or expects or is expected to use, the product. A product
- * is a consumer product regardless of whether the product has substantial
- * commercial, industrial or non-consumer uses, unless such uses represent
- * the only significant mode of use of the product.
- *
- * "Installation Information" for a User Product means any methods,
- * procedures, authorization keys, or other information required to install

* and execute modified versions of a covered work in that User Product from
 * a modified version of its Corresponding Source. The information must
 * suffice to ensure that the continued functioning of the modified object
 * code is in no case prevented or interfered with solely because
 * modification has been made.

*
 * If you convey an object code work under this section in, or with, or
 * specifically for use in, a User Product, and the conveying occurs as
 * part of a transaction in which the right of possession and use of the
 * User Product is transferred to the recipient in perpetuity or for a
 * fixed term (regardless of how the transaction is characterized), the
 * Corresponding Source conveyed under this section must be accompanied
 * by the Installation Information. But this requirement does not apply
 * if neither you nor any third party retains the ability to install
 * modified object code on the User Product (for example, the work has
 * been installed in ROM).

*
 * The requirement to provide Installation Information does not include a
 * requirement to continue to provide support service, warranty, or updates
 * for a work that has been modified or installed by the recipient, or for
 * the User Product in which it has been modified or installed. Access to a
 * network may be denied when the modification itself materially and
 * adversely affects the operation of the network or violates the rules and
 * protocols for communication across the network.

*
 * Corresponding Source conveyed, and Installation Information provided,
 * in accord with this section must be in a format that is publicly
 * documented (and with an implementation available to the public in
 * source code form), and must require no special password or key for
 * unpacking, reading or copying.

*
 * 7. Additional Terms.

*
 * "Additional permissions" are terms that supplement the terms of this
 * License by making exceptions from one or more of its conditions.
 * Additional permissions that are applicable to the entire Program shall
 * be treated as though they were included in this License, to the extent
 * that they are valid under applicable law. If additional permissions
 * apply only to part of the Program, that part may be used separately
 * under those permissions, but the entire Program remains governed by
 * this License without regard to the additional permissions.

*
 * When you convey a copy of a covered work, you may at your option
 * remove any additional permissions from that copy, or from any part of
 * it. (Additional permissions may be written to require their own
 * removal in certain cases when you modify the work.) You may place
 * additional permissions on material, added by you to a covered work,
 * for which you have or can give appropriate copyright permission.

*
 * Notwithstanding any other provision of this License, for material you
 * add to a covered work, you may (if authorized by the copyright holders of
 * that material) supplement the terms of this License with terms:

*
 * a) Disclaiming warranty or limiting liability differently from the
 * terms of sections 15 and 16 of this License; or

- *
 - * b) Requiring preservation of specified reasonable legal notices or
 - * author attributions in that material or in the Appropriate Legal
 - * Notices displayed by works containing it; or
 - *
 - * c) Prohibiting misrepresentation of the origin of that material, or
 - * requiring that modified versions of such material be marked in
 - * reasonable ways as different from the original version; or
 - *
 - * d) Limiting the use for publicity purposes of names of licensors or
 - * authors of the material; or
 - *
 - * e) Declining to grant rights under trademark law for use of some
 - * trade names, trademarks, or service marks; or
 - *
 - * f) Requiring indemnification of licensors and authors of that
 - * material by anyone who conveys the material (or modified versions of
 - * it) with contractual assumptions of liability to the recipient, for
 - * any liability that these contractual assumptions directly impose on
 - * those licensors and authors.
 - * All other non-permissive additional terms are considered "further
 - * restrictions" within the meaning of section 10. If the Program as you
 - * received it, or any part of it, contains a notice stating that it is
 - * governed by this License along with a term that is a further
 - * restriction, you may remove that term. If a license document contains
 - * a further restriction but permits relicensing or conveying under this
 - * License, you may add to a covered work material governed by the terms
 - * of that license document, provided that the further restriction does
 - * not survive such relicensing or conveying.
 - *
 - * If you add terms to a covered work in accord with this section, you
 - * must place, in the relevant source files, a statement of the
 - * additional terms that apply to those files, or a notice indicating
 - * where to find the applicable terms.
 - *
 - * Additional terms, permissive or non-permissive, may be stated in the
 - * form of a separately written license, or stated as exceptions;
 - * the above requirements apply either way.
 - * 8. Termination.
 - *
 - * You may not propagate or modify a covered work except as expressly
 - * provided under this License. Any attempt otherwise to propagate or
 - * modify it is void, and will automatically terminate your rights under
 - * this License (including any patent licenses granted under the third
 - * paragraph of section 11).
 - *
 - * However, if you cease all violation of this License, then your
 - * license from a particular copyright holder is reinstated (a)
 - * provisionally, unless and until the copyright holder explicitly and
 - * finally terminates your license, and (b) permanently, if the copyright
 - * holder fails to notify you of the violation by some reasonable means
 - * prior to 60 days after the cessation.

- *
 - * Moreover, your license from a particular copyright holder is
 - * reinstated permanently if the copyright holder notifies you of the
 - * violation by some reasonable means, this is the first time you have
 - * received notice of violation of this License (for any work) from that
 - * copyright holder, and you cure the violation prior to 30 days after
 - * your receipt of the notice.
- *
 - * Termination of your rights under this section does not terminate the
 - * licenses of parties who have received copies or rights from you under
 - * this License. If your rights have been terminated and not permanently
 - * reinstated, you do not qualify to receive new licenses for the same
 - * material under section 10.
- *
 - * 9. Acceptance Not Required for Having Copies.
- *
 - * You are not required to accept this License in order to receive or
 - * run a copy of the Program. Ancillary propagation of a covered work
 - * occurring solely as a consequence of using peer-to-peer transmission
 - * to receive a copy likewise does not require acceptance. However,
 - * nothing other than this License grants you permission to propagate or
 - * modify any covered work. These actions infringe copyright if you do
 - * not accept this License. Therefore, by modifying or propagating a
 - * covered work, you indicate your acceptance of this License to do so.
- *
 - * 10. Automatic Licensing of Downstream Recipients.
- *
 - * Each time you convey a covered work, the recipient automatically
 - * receives a license from the original licensors, to run, modify and
 - * propagate that work, subject to this License. You are not responsible
 - * for enforcing compliance by third parties with this License.
- *
 - * An "entity transaction" is a transaction transferring control of an
 - * organization, or substantially all assets of one, or subdividing an
 - * organization, or merging organizations. If propagation of a covered
 - * work results from an entity transaction, each party to that
 - * transaction who receives a copy of the work also receives whatever
 - * licenses to the work the party's predecessor in interest had or could
 - * give under the previous paragraph, plus a right to possession of the
 - * Corresponding Source of the work from the predecessor in interest, if
 - * the predecessor has it or can get it with reasonable efforts.
- *
 - * You may not impose any further restrictions on the exercise of the
 - * rights granted or affirmed under this License. For example, you may
 - * not impose a license fee, royalty, or other charge for exercise of
 - * rights granted under this License, and you may not initiate litigation
 - * (including a cross-claim or counterclaim in a lawsuit) alleging that
 - * any patent claim is infringed by making, using, selling, offering for
 - * sale, or importing the Program or any portion of it.
- *
 - * 11. Patents.
- *
 - * A "contributor" is a copyright holder who authorizes use under this
 - * License of the Program or a work on which the Program is based. The
 - * work thus licensed is called the contributor's "contributor version".

- *
 - * A contributor's "essential patent claims" are all patent claims
 - * owned or controlled by the contributor, whether already acquired or
 - * hereafter acquired, that would be infringed by some manner, permitted
 - * by this License, of making, using, or selling its contributor version,
 - * but do not include claims that would be infringed only as a
 - * consequence of further modification of the contributor version. For
 - * purposes of this definition, "control" includes the right to grant
 - * patent sublicenses in a manner consistent with the requirements of
 - * this License.
- *
 - * Each contributor grants you a non-exclusive, worldwide, royalty-free
 - * patent license under the contributor's essential patent claims, to
 - * make, use, sell, offer for sale, import and otherwise run, modify and
 - * propagate the contents of its contributor version.
- *
 - * In the following three paragraphs, a "patent license" is any express
 - * agreement or commitment, however denominated, not to enforce a patent
 - * (such as an express permission to practice a patent or covenant not to
 - * sue for patent infringement). To "grant" such a patent license to a
 - * party means to make such an agreement or commitment not to enforce a
 - * patent against the party.
- *
 - * If you convey a covered work, knowingly relying on a patent license,
 - * and the Corresponding Source of the work is not available for anyone
 - * to copy, free of charge and under the terms of this License, through a
 - * publicly available network server or other readily accessible means,
 - * then you must either (1) cause the Corresponding Source to be so
 - * available, or (2) arrange to deprive yourself of the benefit of the
 - * patent license for this particular work, or (3) arrange, in a manner
 - * consistent with the requirements of this License, to extend the patent
 - * license to downstream recipients. "Knowingly relying" means you have
 - * actual knowledge that, but for the patent license, your conveying the
 - * covered work in a country, or your recipient's use of the covered work
 - * in a country, would infringe one or more identifiable patents in that
 - * country that you have reason to believe are valid.
- *
 - * If, pursuant to or in connection with a single transaction or
 - * arrangement, you convey, or propagate by procuring conveyance of, a
 - * covered work, and grant a patent license to some of the parties
 - * receiving the covered work authorizing them to use, propagate, modify
 - * or convey a specific copy of the covered work, then the patent license
 - * you grant is automatically extended to all recipients of the covered
 - * work and works based on it.
- *
 - * A patent license is "discriminatory" if it does not include within
 - * the scope of its coverage, prohibits the exercise of, or is
 - * conditioned on the non-exercise of one or more of the rights that are
 - * specifically granted under this License. You may not convey a covered
 - * work if you are a party to an arrangement with a third party that is
 - * in the business of distributing software, under which you make payment
 - * to the third party based on the extent of your activity of conveying
 - * the work, and under which the third party grants, to any of the
 - * parties who would receive the covered work from you, a discriminatory

- * patent license (a) in connection with copies of the covered work
- * conveyed by you (or copies made from those copies), or (b) primarily
- * for and in connection with specific products or compilations that
- * contain the covered work, unless you entered into that arrangement,
- * or that patent license was granted, prior to 28 March 2007.
- *
- * Nothing in this License shall be construed as excluding or limiting
- * any implied license or other defenses to infringement that may
- * otherwise be available to you under applicable patent law.
- *
- * 12. No Surrender of Others' Freedom.
- *
- * If conditions are imposed on you (whether by court order, agreement or
- * otherwise) that contradict the conditions of this License, they do not
- * excuse you from the conditions of this License. If you cannot convey a
- * covered work so as to satisfy simultaneously your obligations under this
- * License and any other pertinent obligations, then as a consequence you may
- * not convey it at all. For example, if you agree to terms that obligate you
- * to collect a royalty for further conveying from those to whom you convey
- * the Program, the only way you could satisfy both those terms and this
- * License would be to refrain entirely from conveying the Program.
- *
- * 13. Remote Network Interaction; Use with the GNU General Public License.
- *
- * Notwithstanding any other provision of this License, if you modify the
- * Program, your modified version must prominently offer all users
- * interacting with it remotely through a computer network (if your version
- * supports such interaction) an opportunity to receive the Corresponding
- * Source of your version by providing access to the Corresponding Source
- * from a network server at no charge, through some standard or customary
- * means of facilitating copying of software. This Corresponding Source
- * shall include the Corresponding Source for any work covered by version 3
- * of the GNU General Public License that is incorporated pursuant to the
- * following paragraph.
- *
- * Notwithstanding any other provision of this License, you have
- * permission to link or combine any covered work with a work licensed
- * under version 3 of the GNU General Public License into a single
- * combined work, and to convey the resulting work. The terms of this
- * License will continue to apply to the part which is the covered work,
- * but the work with which it is combined will remain governed by version
- * 3 of the GNU General Public License.
- *
- * 14. Revised Versions of this License.
- *
- * The Free Software Foundation may publish revised and/or new versions of
- * the GNU Affero General Public License from time to time. Such new versions
- * will be similar in spirit to the present version, but may differ in detail to
- * address new problems or concerns.
- *
- * Each version is given a distinguishing version number. If the
- * Program specifies that a certain numbered version of the GNU Affero General
- * Public License "or any later version" applies to it, you have the
- * option of following the terms and conditions either of that numbered
- * version or of any later version published by the Free Software

* Foundation. If the Program does not specify a version number of the
 * GNU Affero General Public License, you may choose any version ever published
 * by the Free Software Foundation.

*
 * If the Program specifies that a proxy can decide which future
 * versions of the GNU Affero General Public License can be used, that proxy's
 * public statement of acceptance of a version permanently authorizes you
 * to choose that version for the Program.

*
 * Later license versions may give you additional or different
 * permissions. However, no additional obligations are imposed on any
 * author or copyright holder as a result of your choosing to follow a
 * later version.

*
 * 15. Disclaimer of Warranty.

*
 * THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
 * APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
 * HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
 * OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
 * IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
 * ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

*
 * 16. Limitation of Liability.

*
 * IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
 * WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
 * THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
 * GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
 * USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
 * DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
 * PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
 * EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGES.

*
 * 17. Interpretation of Sections 15 and 16.

*
 * If the disclaimer of warranty and limitation of liability provided
 * above cannot be given local legal effect according to their terms,
 * reviewing courts shall apply local law that most closely approximates
 * an absolute waiver of all civil liability in connection with the
 * Program, unless a warranty or assumption of liability accompanies a
 * copy of the Program in return for a fee.

*
 *
 * END OF TERMS AND CONDITIONS

*
 *
 * How to Apply These Terms to Your New Programs

*
 * If you develop a new program, and you want it to be of the greatest
 * possible use to the public, the best way to achieve this is to make it
 * free software which everyone can redistribute and change under these terms.
 *

*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.

*/

/*

* Copyright (c) 1995, 1996

*The President and Fellows of Harvard University. All rights reserved.

*

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:

- * 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
- * 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
- * 3. Neither the name of the University nor the names of its contributors
* may be used to endorse or promote products derived from this software
* without specific prior written permission.

*

* THIS SOFTWARE IS PROVIDED BY HARVARD AND ITS CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL HARVARD OR ITS CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.

*/

=====

/****

* ASM: a very small and fast Java bytecode manipulation framework

* Copyright (c) 2000-2005 INRIA, France Telecom

* All rights reserved.

*

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:

- * 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
- * 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the

- * documentation and/or other materials provided with the distribution.
- * 3. Neither the name of the copyright holders nor the names of its
- * contributors may be used to endorse or promote products derived from
- * this software without specific prior written permission.
- *
- * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
- * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
- * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
- * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
- * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
- * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
- * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
- * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
- * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
- * THE POSSIBILITY OF SUCH DAMAGE.
- */

1.24 dhcp

1.24.1 Available under license

```
# Copyright (c) 2004-2016 by Internet Systems Consortium, Inc. ("ISC")
# Copyright (c) 1995-2003 by Internet Software Consortium
#
# Permission to use, copy, modify, and distribute this software for any
# purpose with or without fee is hereby granted, provided that the above
# copyright notice and this permission notice appear in all copies.
#
# THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES
# WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
# MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR
# ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
# WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
# ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
# OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
#
# Internet Systems Consortium, Inc.
# 950 Charter Street
# Redwood City, CA 94063
# <info@isc.org>
# https://www.isc.org/
```

See the specific source files for any additional copyright or license statements.

1.25 diffutils

1.25.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.26 dpkg

1.26.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.27 e2fsprogs

1.27.1 Available under license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, and the entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ALL OF WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF NOT ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```
/*
 * e2p.h --- header file for the e2p library
 *
 * %Begin-Header%
 * This file may be redistributed under the terms of the GNU Library
 * General Public License, version 2.
 * %End-Header%
 */

#include <sys/types.h> /* Needed by dirent.h on netbsd */
#include <stdio.h>
#include <dirent.h>

#include <ext2fs/ext2_fs.h>

#define E2P_FEATURE_COMPAT0
#define E2P_FEATURE_INCOMPAT1
#define E2P_FEATURE_RO_INCOMPAT2
#define E2P_FEATURE_TYPE_MASK0x03

#define E2P_FEATURE_NEGATE_FLAG0x80

#define E2P_FS_FEATURE0
```



```

#define E2P_JOURNAL_FEATURE1

/* `options' for print_flags() */

#define PFOPT_LONG 1 /* Must be 1 for compatibility with `int long_format'. */

int fgetflags (const char * name, unsigned long * flags);
int fgetversion (const char * name, unsigned long * version);
int fsetflags (const char * name, unsigned long flags);
int fsetversion (const char * name, unsigned long version);
int fgetproject(const char *name, unsigned long *project);
int fsetproject(const char *name, unsigned long project);
int getflags (int fd, unsigned long * flags);
int getversion (int fd, unsigned long * version);
int iterate_on_dir (const char * dir_name,
    int (*func) (const char *, struct dirent *, void *),
    void * private_arg);
void list_super(struct ext2_super_block * s);
void list_super2(struct ext2_super_block * s, FILE *f);
void print_fs_errors (FILE * f, unsigned short errors);
void print_flags (FILE * f, unsigned long flags, unsigned options);
void print_fs_state (FILE * f, unsigned short state);
int setflags (int fd, unsigned long flags);
int setversion (int fd, unsigned long version);

const char *e2p_feature2string(int compat, unsigned int mask);
const char *e2p_jrnl_feature2string(int compat, unsigned int mask);
int e2p_string2feature(char *string, int *compat, unsigned int *mask);
int e2p_jrnl_string2feature(char *string, int *compat_type, unsigned int *mask);
int e2p_edit_feature(const char *str, __u32 *compat_array, __u32 *ok_array);
int e2p_edit_feature2(const char *str, __u32 *compat_array, __u32 *ok_array,
    __u32 *clear_ok_array, int *type_err,
    unsigned int *mask_err);

int e2p_is_null_uuid(void *uu);
void e2p_uuid_to_str(void *uu, char *out);
const char *e2p_uuid2str(void *uu);

const char *e2p_hash2string(int num);
int e2p_string2hash(char *string);

const char *e2p_mntopt2string(unsigned int mask);
int e2p_string2mntopt(char *string, unsigned int *mask);
int e2p_edit_mntopts(const char *str, __u32 *mntopts, __u32 ok);

unsigned long parse_num_blocks(const char *arg, int log_block_size);
unsigned long long parse_num_blocks2(const char *arg, int log_block_size);

char *e2p_os2string(int os_type);
int e2p_string2os(char *str);

unsigned int e2p_percent(int percent, unsigned int base);
/*

```

```

* Copyright 1987 by MIT Student Information Processing Board
*
* Permission to use, copy, modify, and distribute this software and
* its documentation for any purpose is hereby granted, provided that
* the names of M.I.T. and the M.I.T. S.I.P.B. not be used in
* advertising or publicity pertaining to distribution of the software
* without specific, written prior permission. M.I.T. and the
* M.I.T. S.I.P.B. make no representations about the suitability of
* this software for any purpose. It is provided "as is" without
* express or implied warranty.
*/

#include "config.h"
#include "com_err.h"
#include "error_table.h"
#include "internal.h"

static const char char_set[] =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789_";

static char buf[6];

const char * error_table_name(errcode_t num)
{
    int ch;
    int i;
    char *p;

    /* num = aa aaa abb bbb bcc ccc cdd ddd d?? ??? ??? */
    p = buf;
    num >>= ERRCODE_RANGE;
    /* num = ?? ??? ??? aaa aaa bbb bbb ccc ccc ddd ddd */
    num &= 077777777L;
    /* num = 00 000 000 aaa aaa bbb bbb ccc ccc ddd ddd */
    for (i = 4; i >= 0; i--) {
        ch = (int)((num >> BITS_PER_CHAR * i) & ((1 << BITS_PER_CHAR) - 1));
        if (ch != 0)
            *p++ = char_set[ch-1];
    }
    *p = '\0';
    return(buf);
}
/*
* ext2fs.h --- ext2fs
*
* Copyright (C) 1993, 1994, 1995, 1996 Theodore Ts'o.
*
* %Begin-Header%
* This file may be redistributed under the terms of the GNU Library
* General Public License, version 2.
* %End-Header%
*/

#ifndef _EXT2FS_EXT2FS_H
#define _EXT2FS_EXT2FS_H

```

```

#ifdef __GNUC__
#define EXT2FS_ATTR(x) __attribute__(x)
#else
#define EXT2FS_ATTR(x)
#endif

#ifdef __cplusplus
extern "C" {
#endif

/*
 * Non-GNU C compilers won't necessarily understand inline
 */
#if (!defined(__GNUC__) && !defined(__WATCOMC__))
#define NO_INLINE_FUNCS
#endif

/*
 * Where the master copy of the superblock is located, and how big
 * superblocks are supposed to be. We define SUPERBLOCK_SIZE because
 * the size of the superblock structure is not necessarily trustworthy
 * (some versions have the padding set up so that the superblock is
 * 1032 bytes long).
 */
#define SUPERBLOCK_OFFSET1024
#define SUPERBLOCK_SIZE1024

#define UUID_STR_SIZE 37

/*
 * The last ext2fs revision level that this version of the library is
 * able to support.
 */
#define EXT2_LIB_CURRENT_REVEXT2_DYNAMIC_REV

#ifdef HAVE_SYS_TYPES_H
#include <sys/types.h>
#endif

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>

#if EXT2_FLAT_INCLUDES
#include "e2_types.h"
#include "ext2_fs.h"
#include "ext3_extents.h"
#else
#include <ext2fs/ext2_types.h>
#include <ext2fs/ext2_fs.h>
#include <ext2fs/ext3_extents.h>
#endif /* EXT2_FLAT_INCLUDES */

```

```

typedef __u32 __bitwiseext2_ino_t;
typedef __u32 __bitwiseblk_t;
typedef __u64 __bitwiseblk64_t;
typedef __u32 __bitwisedgrp_t;
typedef __u32 __bitwiseext2_off_t;
typedef __u64 __bitwiseext2_off64_t;
typedef __s64 __bitwisee2_blkcnt_t;
typedef __u32 __bitwiseext2_dirhash_t;

#if EXT2_FLAT_INCLUDES
#include "com_err.h"
#include "ext2_io.h"
#include "ext2_err.h"
#include "ext2_ext_attr.h"
#else
#include <et/com_err.h>
#include <ext2fs/ext2_io.h>
#include <ext2fs/ext2_err.h>
#include <ext2fs/ext2_ext_attr.h>
#endif

/*
 * Portability help for Microsoft Visual C++
 */
#ifndef _MSC_VER
#define EXT2_QSORT_TYPE int __cdecl
#else
#define EXT2_QSORT_TYPE int
#endif

typedef struct struct_ext2_filsys *ext2_filsys;

#define EXT2FS_MARK_ERROR 0
#define EXT2FS_UNMARK_ERROR 1
#define EXT2FS_TEST_ERROR2

typedef struct ext2fs_struct_generic_bitmap *ext2fs_generic_bitmap;
typedef struct ext2fs_struct_generic_bitmap *ext2fs_inode_bitmap;
typedef struct ext2fs_struct_generic_bitmap *ext2fs_block_bitmap;

#define EXT2_FIRST_INODE(s)EXT2_FIRST_INO(s)

/*
 * Badblocks list definitions
 */

typedef struct ext2_struct_u32_list *ext2_badblocks_list;
typedef struct ext2_struct_u32_iterate *ext2_badblocks_iterate;

typedef struct ext2_struct_u32_list *ext2_u32_list;
typedef struct ext2_struct_u32_iterate *ext2_u32_iterate;

/* old */

```

```
typedef struct ext2_struct_u32_list *badblocks_list;
typedef struct ext2_struct_u32_iterate *badblocks_iterate;

#define BADBLOCKS_FLAG_DIRTY1

/*
 * ext2_dblist structure and abstractions (see dblist.c)
 */
struct ext2_db_entry2 {
    ext2_ino_tino;
    blk64_tblk;
    e2_blkcnt_tblockcnt;
};

/* Ye Olde 32-bit version */
struct ext2_db_entry {
    ext2_ino_tino;
    blk_tblk;
    intblockcnt;
};

typedef struct ext2_struct_dblist *ext2_dblist;

#define DBLIST_ABORT1

/*
 * ext2_fileio definitions
 */

#define EXT2_FILE_WRITE0x0001
#define EXT2_FILE_CREATE0x0002

#define EXT2_FILE_MASK0x00FF

#define EXT2_FILE_BUF_DIRTY0x4000
#define EXT2_FILE_BUF_VALID0x2000

typedef struct ext2_file *ext2_file_t;

#define EXT2_SEEK_SET0
#define EXT2_SEEK_CUR1
#define EXT2_SEEK_END2

/*
 * Flags for the ext2_filsys structure and for ext2fs_open()
 */
#define EXT2_FLAG_RW0x01
#define EXT2_FLAG_CHANGED0x02
#define EXT2_FLAG_DIRTY0x04
#define EXT2_FLAG_VALID0x08
#define EXT2_FLAG_IB_DIRTY0x10
#define EXT2_FLAG_BB_DIRTY0x20
#define EXT2_FLAG_SWAP_BYTES0x40
#define EXT2_FLAG_SWAP_BYTES_READ0x80
```

```

#define EXT2_FLAG_SWAP_BYTES_WRITE0x100
#define EXT2_FLAG_MASTER_SB_ONLY0x200
#define EXT2_FLAG_FORCE0x400
#define EXT2_FLAG_SUPER_ONLY0x800
#define EXT2_FLAG_JOURNAL_DEV_OK0x1000
#define EXT2_FLAG_IMAGE_FILE0x2000
#define EXT2_FLAG_EXCLUSIVE0x4000
#define EXT2_FLAG_SOFTSUPP_FEATURES0x8000
#define EXT2_FLAG_NOFREE_ON_ERROR0x10000
#define EXT2_FLAG_64BITS0x20000
#define EXT2_FLAG_PRINT_PROGRESS0x40000
#define EXT2_FLAG_DIRECT_IO0x80000
#define EXT2_FLAG_SKIP_MMP0x100000
#define EXT2_FLAG_IGNORE_CSUM_ERRORS0x200000

/*
 * Special flag in the ext2 inode i_flag field that means that this is
 * a new inode. (So that ext2_write_inode() can clear extra fields.)
 */
#define EXT2_NEW_INODE_FL0x80000000

/*
 * Flags for mkjournal
 */
#define EXT2_MKJOURNAL_V1_SUPER0x00000001 /* create V1 superblock (deprecated) */
#define EXT2_MKJOURNAL_LAZYINIT0x00000002 /* don't zero journal inode before use */
#define EXT2_MKJOURNAL_NO_MNT_CHECK 0x00000004 /* don't check mount status */

struct opaque_ext2_group_desc;

struct struct_ext2_filsys {
    errcode_tmagic;
    io_channelio;
    intflags;
    char *device_name;
    struct ext2_super_block* super;
    unsigned intblocksize;
    intfragsize;
    dgrp_tgroup_desc_count;
    unsigned longdesc_blocks;
    struct opaque_ext2_group_desc *group_desc;
    unsigned intinode_blocks_per_group;
    ext2fs_inode_bitmapinode_map;
    ext2fs_block_bitmapblock_map;
    /* XXX FIXME-64: not 64-bit safe, but not used? */
    errcode_t (*get_blocks)(ext2_filsys fs, ext2_ino_t ino, blk_t *blocks);
    errcode_t (*check_directory)(ext2_filsys fs, ext2_ino_t ino);
    errcode_t (*write_bitmaps)(ext2_filsys fs);
    errcode_t (*read_inode)(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode);
    errcode_t (*write_inode)(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode);
    ext2_badblocks_listbadblocks;
    ext2_dblistdblist;
    __u32stride; /* for mke2fs */

```

```

struct ext2_super_block *orig_super;
struct ext2_image_hdr *image_header;
__u32umask;
time_tnow;
intcluster_ratio_bits;
__u16default_bitmap_type;
__u16pad;
/*
 * Reserved for future expansion
 */
__u32reserved[5];

/*
 * Reserved for the use of the calling application.
 */
void *priv_data;

/*
 * Inode cache
 */
struct ext2_inode_cache*icache;
io_channelimage_io;

/*
 * More callback functions
 */
errcode_t (*get_alloc_block)(ext2_filsys fs, blk64_t goal,
    blk64_t *ret);
void (*block_alloc_stats)(ext2_filsys fs, blk64_t blk, int inuse);

/*
 * Buffers for Multiple mount protection(MMP) block.
 */
void *mmp_buf;
void *mmp_cmp;
int mmp_fd;

/*
 * Time at which e2fsck last updated the MMP block.
 */
long mmp_last_written;

/* progress operation functions */
struct ext2fs_progress_ops *progress_ops;

/* Precomputed FS UUID checksum for seeding other checksums */
__u32 csum_seed;

io_channeljournal_io;
char*journal_name;

/* New block range allocation hooks */
errcode_t (*new_range)(ext2_filsys fs, int flags, blk64_t goal,
    blk64_t len, blk64_t *pblk, blk64_t *plen);

```

```

void (*block_alloc_stats_range)(ext2_filsys fs, blk64_t blk, blk_t num,
int inuse);
};

#ifdef EXT2_FLAT_INCLUDES
#include "e2_bitops.h"
#else
#include <ext2fs/bitops.h>
#endif

/*
 * 64-bit bitmap backend types
 */
#define EXT2FS_BMAP64_BITARRAY1
#define EXT2FS_BMAP64_RBTREE2
#define EXT2FS_BMAP64_AUTODIR3

/*
 * Return flags for the block iterator functions
 */
#define BLOCK_CHANGED1
#define BLOCK_ABORT2
#define BLOCK_ERROR4
#define BLOCK_INLINE_DATA_CHANGED8

/*
 * Block iterate flags
 *
 * BLOCK_FLAG_APPEND, or BLOCK_FLAG_HOLE, indicates that the iterator
 * function should be called on blocks where the block number is zero.
 * This is used by ext2fs_expand_dir() to be able to add a new block
 * to an inode. It can also be used for programs that want to be able
 * to deal with files that contain "holes".
 *
 * BLOCK_FLAG_DEPTH_TRAVERSE indicates that the iterator function for
 * the indirect, doubly indirect, etc. blocks should be called after
 * all of the blocks contained in the indirect blocks are processed.
 * This is useful if you are going to be deallocating blocks from an
 * inode.
 *
 * BLOCK_FLAG_DATA_ONLY indicates that the iterator function should be
 * called for data blocks only.
 *
 * BLOCK_FLAG_READ_ONLY is a promise by the caller that it will not
 * modify returned block number.
 *
 * BLOCK_FLAG_NO_LARGE is for internal use only. It informs
 * ext2fs_block_iterate2 that large files won't be accepted.
 */
#define BLOCK_FLAG_APPEND1
#define BLOCK_FLAG_HOLE1
#define BLOCK_FLAG_DEPTH_TRAVERSE2
#define BLOCK_FLAG_DATA_ONLY4
#define BLOCK_FLAG_READ_ONLY8

```



```

#define BLOCK_FLAG_NO_LARGE 0x1000

/*
 * Magic "block count" return values for the block iterator function.
 */
#define BLOCK_COUNT_IND(-1)
#define BLOCK_COUNT_DIND(-2)
#define BLOCK_COUNT_TIND(-3)
#define BLOCK_COUNT_TRANSLATOR(-4)

#if 0
/*
 * Flags for ext2fs_move_blocks
 */
#define EXT2_BMOVE_GET_DBLIST 0x0001
#define EXT2_BMOVE_DEBUG 0x0002
#endif

/*
 * Generic (non-filesystem layout specific) extents structure
 */

#define EXT2_EXTENT_FLAGS_LEAF 0x0001
#define EXT2_EXTENT_FLAGS_UNINIT 0x0002
#define EXT2_EXTENT_FLAGS_SECOND_VISIT 0x0004

struct ext2fs_extent {
    blk64_t te_pblk; /* first physical block */
    blk64_t te_lblk; /* first logical block extent covers */
    __u32 te_len; /* number of blocks covered by extent */
    __u32 te_flags; /* extent flags */
};

typedef struct ext2_extent_handle *ext2_extent_handle_t;
typedef struct ext2_extent_path *ext2_extent_path_t;

/*
 * Flags used by ext2fs_extent_get()
 */
#define EXT2_EXTENT_CURRENT 0x0000
#define EXT2_EXTENT_MOVE_MASK 0x000F
#define EXT2_EXTENT_ROOT 0x0001
#define EXT2_EXTENT_LAST_LEAF 0x0002
#define EXT2_EXTENT_FIRST_SIB 0x0003
#define EXT2_EXTENT_LAST_SIB 0x0004
#define EXT2_EXTENT_NEXT_SIB 0x0005
#define EXT2_EXTENT_PREV_SIB 0x0006
#define EXT2_EXTENT_NEXT_LEAF 0x0007
#define EXT2_EXTENT_PREV_LEAF 0x0008
#define EXT2_EXTENT_NEXT 0x0009
#define EXT2_EXTENT_PREV 0x000A
#define EXT2_EXTENT_UP 0x000B
#define EXT2_EXTENT_DOWN 0x000C
#define EXT2_EXTENT_DOWN_AND_LAST 0x000D

```

```

/*
 * Flags used by ext2fs_extent_insert()
 */
#define EXT2_EXTENT_INSERT_AFTER0x0001 /* insert after handle loc'n */
#define EXT2_EXTENT_INSERT_NOSPLIT0x0002 /* insert may not cause split */

/*
 * Flags used by ext2fs_extent_delete()
 */
#define EXT2_EXTENT_DELETE_KEEP_EMPTY0x001 /* keep node if last extnt gone */

/*
 * Flags used by ext2fs_extent_set_bmap()
 */
#define EXT2_EXTENT_SET_BMAP_UNINIT0x0001

/*
 * Data structure returned by ext2fs_extent_get_info()
 */
struct ext2_extent_info {
    intcurr_entry;
    intcurr_level;
    intnum_entries;
    intmax_entries;
    intmax_depth;
    intbytes_avail;
    blk64_tmax_lblk;
    blk64_tmax_pblk;
    __u32max_len;
    __u32max_uninit_len;
};

/*
 * Flags for directory block reading and writing functions
 */
#define EXT2_DIRBLOCK_V2_STRUCT0x0001

/*
 * Return flags for the directory iterator functions
 */
#define DIRENT_CHANGED1
#define DIRENT_ABORT2
#define DIRENT_ERROR3

/*
 * Directory iterator flags
 */

#define DIRENT_FLAG_INCLUDE_EMPTY1
#define DIRENT_FLAG_INCLUDE_REMOVED2
#define DIRENT_FLAG_INCLUDE_CSUM4
#define DIRENT_FLAG_INCLUDE_INLINE_DATA 8

#define DIRENT_DOT_FILE1

```

```
#define DIRENT_DOT_DOT_FILE2
#define DIRENT_OTHER_FILE3
#define DIRENT_DELETED_FILE4
#define DIRENT_CHECKSUM5

/*
 * Inode scan definitions
 */
typedef struct ext2_struct_inode_scan *ext2_inode_scan;

/*
 * ext2fs_scan flags
 */
#define EXT2_SF_CHK_BADBLOCKS0x0001
#define EXT2_SF_BAD_INODE_BLK0x0002
#define EXT2_SF_BAD_EXTRA_BYTES0x0004
#define EXT2_SF_SKIP_MISSING_ITABLE0x0008
#define EXT2_SF_DO_LAZY0x0010
#define EXT2_SF_WARN_GARBAGE_INODES0x0020

/*
 * ext2fs_check_if_mounted flags
 */
#define EXT2_MF_MOUNTED1
#define EXT2_MF_ISROOT2
#define EXT2_MF_READONLY4
#define EXT2_MF_SWAP8
#define EXT2_MF_BUSY16

/*
 * Ext2/linux mode flags. We define them here so that we don't need
 * to depend on the OS's sys/stat.h, since we may be compiling on a
 * non-Linux system.
 */
#define LINUX_S_IFMT 00170000
#define LINUX_S_IFSOCK 0140000
#define LINUX_S_IFLNK 0120000
#define LINUX_S_IFREG 0100000
#define LINUX_S_IFBLK 0060000
#define LINUX_S_IFDIR 0040000
#define LINUX_S_IFCHR 0020000
#define LINUX_S_FIFO 0010000
#define LINUX_S_ISUID 0004000
#define LINUX_S_ISGID 0002000
#define LINUX_S_ISVTX 0001000

#define LINUX_S_IRWXU 00700
#define LINUX_S_IRUSR 00400
#define LINUX_S_IWUSR 00200
#define LINUX_S_IXUSR 00100

#define LINUX_S_IRWXG 00070
#define LINUX_S_IRGRP 00040
#define LINUX_S_IWGRP 00020
```

```

#define LINUX_S_IXGRP 00010

#define LINUX_S_IRWXO 00007
#define LINUX_S_IROTH 00004
#define LINUX_S_IWOTH 00002
#define LINUX_S_IXOTH 00001

#define LINUX_S_ISLNK(m)((m) & LINUX_S_IFMT) == LINUX_S_IFLNK)
#define LINUX_S_ISREG(m)((m) & LINUX_S_IFMT) == LINUX_S_IFREG)
#define LINUX_S_ISDIR(m)((m) & LINUX_S_IFMT) == LINUX_S_IFDIR)
#define LINUX_S_ISCHR(m)((m) & LINUX_S_IFMT) == LINUX_S_IFCHR)
#define LINUX_S_ISBLK(m)((m) & LINUX_S_IFMT) == LINUX_S_IFBLK)
#define LINUX_S_ISFIFO(m)((m) & LINUX_S_IFMT) == LINUX_S_IFIFO)
#define LINUX_S_ISSOCK(m)((m) & LINUX_S_IFMT) == LINUX_S_IFSOCK)

/*
 * ext2 size of an inode
 */
#define EXT2_I_SIZE(i)((i)->i_size | ((__u64) (i)->i_size_high << 32))

/*
 * ext2_icount_t abstraction
 */
#define EXT2_ICOUNT_OPT_INCREMENT 0x01

typedef struct ext2_icount *ext2_icount_t;

/*
 * Flags for ext2fs_bmap
 */
#define BMAP_ALLOC 0x0001
#define BMAP_SET 0x0002
#define BMAP_UNINIT 0x0004
#define BMAP_ZERO 0x0008

/*
 * Returned flags from ext2fs_bmap
 */
#define BMAP_RET_UNINIT 0x0001

/*
 * Flags for imager.c functions
 */
#define IMAGER_FLAG_INODEMAP 1
#define IMAGER_FLAG_SPARSEWRITE 2

/*
 * For checking structure magic numbers...
 */

#define EXT2_CHECK_MAGIC(struct, code) \
    if ((struct)->magic != (code)) return (code)

/*
 * Features supported by this version of the library

```

```

*/
#define EXT2_LIB_FEATURE_COMPAT_SUPP(EXT2_FEATURE_COMPAT_DIR_PREALLOC\
EXT2_FEATURE_COMPAT_IMAGIC_INODES\
EXT3_FEATURE_COMPAT_HAS_JOURNAL\
EXT2_FEATURE_COMPAT_RESIZE_INODE\
EXT2_FEATURE_COMPAT_DIR_INDEX\
EXT2_FEATURE_COMPAT_EXT_ATTR\
EXT4_FEATURE_COMPAT_SPARSE_SUPER2)

#ifdef CONFIG_MMP
#define EXT4_LIB_INCOMPAT_MMPEXT4_FEATURE_INCOMPAT_MMP
#else
#define EXT4_LIB_INCOMPAT_MMP(0)
#endif

#define EXT2_LIB_FEATURE_INCOMPAT_SUPP(EXT2_FEATURE_INCOMPAT_FILETYPE\
EXT3_FEATURE_INCOMPAT_JOURNAL_DEV\
EXT2_FEATURE_INCOMPAT_META_BG\
EXT3_FEATURE_INCOMPAT_RECOVER\
EXT3_FEATURE_INCOMPAT_EXTENTS\
EXT4_FEATURE_INCOMPAT_FLEX_BG\
EXT4_LIB_INCOMPAT_MMP\
EXT4_FEATURE_INCOMPAT_64BIT\
EXT4_FEATURE_INCOMPAT_INLINE_DATA\
EXT4_FEATURE_INCOMPAT_ENCRYPT\
EXT4_FEATURE_INCOMPAT_CSUM_SEED)

#define EXT2_LIB_FEATURE_RO_COMPAT_SUPP(EXT2_FEATURE_RO_COMPAT_SPARSE_SUPER\
EXT4_FEATURE_RO_COMPAT_HUGE_FILE\
EXT2_FEATURE_RO_COMPAT_LARGE_FILE\
EXT4_FEATURE_RO_COMPAT_DIR_NLINK\
EXT4_FEATURE_RO_COMPAT_EXTRA_ISIZE\
EXT4_FEATURE_RO_COMPAT_GDT_CSUM\
EXT4_FEATURE_RO_COMPAT_BIGALLOC\
EXT4_FEATURE_RO_COMPAT_QUOTA\
EXT4_FEATURE_RO_COMPAT_METADATA_CSUM\
EXT4_FEATURE_RO_COMPAT_READONLY\
EXT4_FEATURE_RO_COMPAT_PROJECT)

/*
 * These features are only allowed if EXT2_FLAG_SOFTSUPP_FEATURES is passed
 * to ext2fs_openfs()
 */
#define EXT2_LIB_SOFTSUPP_INCOMPAT(0)
#define EXT2_LIB_SOFTSUPP_RO_COMPAT(EXT4_FEATURE_RO_COMPAT_REPLICA)

/* Translate a block number to a cluster number */
#define EXT2FS_CLUSTER_RATIO(fs)(1 << (fs->cluster_ratio_bits)
#define EXT2FS_CLUSTER_MASK(fs)(EXT2FS_CLUSTER_RATIO(fs) - 1)
#define EXT2FS_B2C(fs, blk)((blk) >> (fs->cluster_ratio_bits)
/* Translate a cluster number to a block number */
#define EXT2FS_C2B(fs, cluster)((cluster) << (fs->cluster_ratio_bits)
/* Translate # of blks to # of clusters */

```

```

#define EXT2FS_NUM_B2C(fs, blks)((blks) + EXT2FS_CLUSTER_MASK(fs)) >> \
(fs)->cluster_ratio_bits)

#if defined(HAVE_FSTAT64) && !defined(__OSX_AVAILABLE_BUT_DEPRECATED)
typedef struct stat64 ext2fs_struct_stat;
#else
typedef struct stat ext2fs_struct_stat;
#endif

/*
 * For ext2fs_close2() and ext2fs_flush2(), this flag allows you to
 * avoid the fsync call.
 */
#define EXT2_FLAG_FLUSH_NO_SYNC      1

/*
 * Modify and iterate extended attributes
 */
struct ext2_xattr_handle;
#define XATTR_ABORT1
#define XATTR_CHANGED2

/*
 * function prototypes
 */
static inline int ext2fs_has_group_desc_csum(ext2_filsys fs)
{
    return ext2fs_has_feature_metadata_csum(fs->super) ||
        ext2fs_has_feature_gdt_csum(fs->super);
}

/* The LARGE_FILE feature should be set if we have stored files 2GB+ in size */
static inline int ext2fs_needs_large_file_feature(unsigned long long file_size)
{
    return file_size >= 0x80000000ULL;
}

/* alloc.c */
extern void ext2fs_clear_block_uninit(ext2_filsys fs, dgrp_t group);
extern errcode_t ext2fs_new_inode(ext2_filsys fs, ext2_ino_t dir, int mode,
    ext2fs_inode_bitmap map, ext2_ino_t *ret);
extern errcode_t ext2fs_new_block(ext2_filsys fs, blk_t goal,
    ext2fs_block_bitmap map, blk_t *ret);
extern errcode_t ext2fs_new_block2(ext2_filsys fs, blk64_t goal,
    ext2fs_block_bitmap map, blk64_t *ret);
extern errcode_t ext2fs_get_free_blocks(ext2_filsys fs, blk_t start,
    blk_t finish, int num,
    ext2fs_block_bitmap map,
    blk_t *ret);
extern errcode_t ext2fs_get_free_blocks2(ext2_filsys fs, blk64_t start,
    blk64_t finish, int num,
    ext2fs_block_bitmap map,
    blk64_t *ret);
extern errcode_t ext2fs_alloc_block(ext2_filsys fs, blk_t goal,
    char *block_buf, blk_t *ret);

```

```

extern errcode_t ext2fs_alloc_block2(ext2_filsys fs, blk64_t goal,
    char *block_buf, blk64_t *ret);
extern void ext2fs_set_alloc_block_callback(ext2_filsys fs,
    errcode_t (*func)(ext2_filsys fs,
        blk64_t goal,
        blk64_t *ret),
    errcode_t (**old)(ext2_filsys fs,
        blk64_t goal,
        blk64_t *ret));
blk64_t ext2fs_find_inode_goal(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode, blk64_t lblk);
extern void ext2fs_set_new_range_callback(ext2_filsys fs,
    errcode_t (*func)(ext2_filsys fs, int flags, blk64_t goal,
        blk64_t len, blk64_t *pblk, blk64_t *plen),
    errcode_t (**old)(ext2_filsys fs, int flags, blk64_t goal,
        blk64_t len, blk64_t *pblk, blk64_t *plen));
extern void ext2fs_set_block_alloc_stats_range_callback(ext2_filsys fs,
    void (*func)(ext2_filsys fs, blk64_t blk,
        blk_t num, int inuse),
    void (**old)(ext2_filsys fs, blk64_t blk,
        blk_t num, int inuse));
#define EXT2_NEWRANGE_FIXED_GOAL(0x1)
#define EXT2_NEWRANGE_MIN_LENGTH(0x2)
#define EXT2_NEWRANGE_ALL_FLAGS(0x3)
errcode_t ext2fs_new_range(ext2_filsys fs, int flags, blk64_t goal,
    blk64_t len, ext2fs_block_bitmap map, blk64_t *pblk,
    blk64_t *plen);
#define EXT2_ALLOCRANGE_FIXED_GOAL(0x1)
#define EXT2_ALLOCRANGE_ZERO_BLOCKS(0x2)
#define EXT2_ALLOCRANGE_ALL_FLAGS(0x3)
errcode_t ext2fs_alloc_range(ext2_filsys fs, int flags, blk64_t goal,
    blk_t len, blk64_t *ret);

/* alloc_sb.c */
extern int ext2fs_reserve_super_and_bgd(ext2_filsys fs,
    dgrp_t group,
    ext2fs_block_bitmap bmap);
extern void ext2fs_set_block_alloc_stats_callback(ext2_filsys fs,
    void (*func)(ext2_filsys fs,
        blk64_t blk,
        int inuse),
    void (**old)(ext2_filsys fs,
        blk64_t blk,
        int inuse));

/* alloc_stats.c */
void ext2fs_inode_alloc_stats(ext2_filsys fs, ext2_ino_t ino, int inuse);
void ext2fs_inode_alloc_stats2(ext2_filsys fs, ext2_ino_t ino,
    int inuse, int isdir);
void ext2fs_block_alloc_stats(ext2_filsys fs, blk_t blk, int inuse);
void ext2fs_block_alloc_stats2(ext2_filsys fs, blk64_t blk, int inuse);
void ext2fs_block_alloc_stats_range(ext2_filsys fs, blk64_t blk,
    blk_t num, int inuse);

```

```

/* alloc_tables.c */
extern errcode_t ext2fs_allocate_tables(ext2_filsys fs);
extern errcode_t ext2fs_allocate_group_table(ext2_filsys fs, dgrp_t group,
    ext2fs_block_bitmap bmap);

/* badblocks.c */
extern errcode_t ext2fs_u32_list_create(ext2_u32_list *ret, int size);
extern errcode_t ext2fs_u32_list_add(ext2_u32_list bb, __u32 blk);
extern int ext2fs_u32_list_find(ext2_u32_list bb, __u32 blk);
extern int ext2fs_u32_list_test(ext2_u32_list bb, blk_t blk);
extern errcode_t ext2fs_u32_list_iterate_begin(ext2_u32_list bb,
    ext2_u32_iterate *ret);
extern int ext2fs_u32_list_iterate(ext2_u32_iterate iter, blk_t *blk);
extern void ext2fs_u32_list_iterate_end(ext2_u32_iterate iter);
extern errcode_t ext2fs_u32_copy(ext2_u32_list src, ext2_u32_list *dest);
extern int ext2fs_u32_list_equal(ext2_u32_list bb1, ext2_u32_list bb2);

extern errcode_t ext2fs_badblocks_list_create(ext2_badblocks_list *ret,
    int size);
extern errcode_t ext2fs_badblocks_list_add(ext2_badblocks_list bb,
    blk_t blk);
extern int ext2fs_badblocks_list_test(ext2_badblocks_list bb,
    blk_t blk);
extern int ext2fs_u32_list_del(ext2_u32_list bb, __u32 blk);
extern void ext2fs_badblocks_list_del(ext2_u32_list bb, __u32 blk);
extern errcode_t
ext2fs_badblocks_list_iterate_begin(ext2_badblocks_list bb,
    ext2_badblocks_iterate *ret);
extern int ext2fs_badblocks_list_iterate(ext2_badblocks_iterate iter,
    blk_t *blk);
extern void ext2fs_badblocks_list_iterate_end(ext2_badblocks_iterate iter);
extern errcode_t ext2fs_badblocks_copy(ext2_badblocks_list src,
    ext2_badblocks_list *dest);
extern int ext2fs_badblocks_equal(ext2_badblocks_list bb1,
    ext2_badblocks_list bb2);
extern int ext2fs_u32_list_count(ext2_u32_list bb);

/* bb_compat */
extern errcode_t badblocks_list_create(badblocks_list *ret, int size);
extern errcode_t badblocks_list_add(badblocks_list bb, blk_t blk);
extern int badblocks_list_test(badblocks_list bb, blk_t blk);
extern errcode_t badblocks_list_iterate_begin(badblocks_list bb,
    badblocks_iterate *ret);
extern int badblocks_list_iterate(badblocks_iterate iter, blk_t *blk);
extern void badblocks_list_iterate_end(badblocks_iterate iter);
extern void badblocks_list_free(badblocks_list bb);

/* bb_inode.c */
extern errcode_t ext2fs_update_bb_inode(ext2_filsys fs,
    ext2_badblocks_list bb_list);

/* bitmaps.c */
extern void ext2fs_free_block_bitmap(ext2fs_block_bitmap bitmap);
extern void ext2fs_free_inode_bitmap(ext2fs_inode_bitmap bitmap);
extern errcode_t ext2fs_copy_bitmap(ext2fs_generic_bitmap src,

```



```

    ext2fs_generic_bitmap *dest);
extern errcode_t ext2fs_write_inode_bitmap(ext2_filsys fs);
extern errcode_t ext2fs_write_block_bitmap (ext2_filsys fs);
extern errcode_t ext2fs_read_inode_bitmap (ext2_filsys fs);
extern errcode_t ext2fs_read_block_bitmap(ext2_filsys fs);
extern errcode_t ext2fs_allocate_block_bitmap(ext2_filsys fs,
    const char *descr,
    ext2fs_block_bitmap *ret);
extern errcode_t ext2fs_allocate_subcluster_bitmap(ext2_filsys fs,
    const char *descr,
    ext2fs_block_bitmap *ret);
extern int ext2fs_get_bitmap_granularity(ext2fs_block_bitmap bitmap);
extern errcode_t ext2fs_allocate_inode_bitmap(ext2_filsys fs,
    const char *descr,
    ext2fs_inode_bitmap *ret);
extern errcode_t ext2fs_fudge_inode_bitmap_end(ext2fs_inode_bitmap bitmap,
    ext2_ino_t end, ext2_ino_t *oend);
extern errcode_t ext2fs_fudge_block_bitmap_end(ext2fs_block_bitmap bitmap,
    blk_t end, blk_t *oend);
extern errcode_t ext2fs_fudge_block_bitmap_end2(ext2fs_block_bitmap bitmap,
    blk64_t end, blk64_t *oend);
extern void ext2fs_clear_inode_bitmap(ext2fs_inode_bitmap bitmap);
extern void ext2fs_clear_block_bitmap(ext2fs_block_bitmap bitmap);
extern errcode_t ext2fs_read_bitmaps(ext2_filsys fs);
extern errcode_t ext2fs_write_bitmaps(ext2_filsys fs);
extern errcode_t ext2fs_resize_inode_bitmap(__u32 new_end, __u32 new_real_end,
    ext2fs_inode_bitmap bmap);
extern errcode_t ext2fs_resize_inode_bitmap2(__u64 new_end,
    __u64 new_real_end,
    ext2fs_inode_bitmap bmap);
extern errcode_t ext2fs_resize_block_bitmap(__u32 new_end, __u32 new_real_end,
    ext2fs_block_bitmap bmap);
extern errcode_t ext2fs_resize_block_bitmap2(__u64 new_end,
    __u64 new_real_end,
    ext2fs_block_bitmap bmap);
extern errcode_t ext2fs_compare_block_bitmap(ext2fs_block_bitmap bm1,
    ext2fs_block_bitmap bm2);
extern errcode_t ext2fs_compare_inode_bitmap(ext2fs_inode_bitmap bm1,
    ext2fs_inode_bitmap bm2);
extern errcode_t ext2fs_set_inode_bitmap_range(ext2fs_inode_bitmap bmap,
    ext2_ino_t start, unsigned int num,
    void *in);
extern errcode_t ext2fs_set_inode_bitmap_range2(ext2fs_inode_bitmap bmap,
    __u64 start, size_t num,
    void *in);
extern errcode_t ext2fs_get_inode_bitmap_range(ext2fs_inode_bitmap bmap,
    ext2_ino_t start, unsigned int num,
    void *out);
extern errcode_t ext2fs_get_inode_bitmap_range2(ext2fs_inode_bitmap bmap,
    __u64 start, size_t num,
    void *out);
extern errcode_t ext2fs_set_block_bitmap_range(ext2fs_block_bitmap bmap,
    blk_t start, unsigned int num,
    void *in);

```

```

extern errcode_t ext2fs_set_block_bitmap_range(ext2fs_block_bitmap bmap,
blk64_t start, size_t num,
void *in);
extern errcode_t ext2fs_get_block_bitmap_range(ext2fs_block_bitmap bmap,
blk_t start, unsigned int num,
void *out);
extern errcode_t ext2fs_get_block_bitmap_range2(ext2fs_block_bitmap bmap,
blk64_t start, size_t num,
void *out);

/* blknum.c */
extern __u32 ext2fs_inode_bitmap_checksum(ext2_filsys fs, dgrp_t group);
extern __u32 ext2fs_block_bitmap_checksum(ext2_filsys fs, dgrp_t group);
extern dgrp_t ext2fs_group_of_blk2(ext2_filsys fs, blk64_t);
extern blk64_t ext2fs_group_first_block2(ext2_filsys fs, dgrp_t group);
extern blk64_t ext2fs_group_last_block2(ext2_filsys fs, dgrp_t group);
extern int ext2fs_group_blocks_count(ext2_filsys fs, dgrp_t group);
extern blk64_t ext2fs_inode_data_blocks2(ext2_filsys fs,
struct ext2_inode *inode);
extern blk64_t ext2fs_inode_i_blocks(ext2_filsys fs,
struct ext2_inode *inode);
extern blk64_t ext2fs_blocks_count(struct ext2_super_block *super);
extern void ext2fs_blocks_count_set(struct ext2_super_block *super,
blk64_t blk);
extern void ext2fs_blocks_count_add(struct ext2_super_block *super,
blk64_t blk);
extern blk64_t ext2fs_r_blocks_count(struct ext2_super_block *super);
extern void ext2fs_r_blocks_count_set(struct ext2_super_block *super,
blk64_t blk);
extern void ext2fs_r_blocks_count_add(struct ext2_super_block *super,
blk64_t blk);
extern blk64_t ext2fs_free_blocks_count(struct ext2_super_block *super);
extern void ext2fs_free_blocks_count_set(struct ext2_super_block *super,
blk64_t blk);
extern void ext2fs_free_blocks_count_add(struct ext2_super_block *super,
blk64_t blk);
/* Block group descriptor accessor functions */
extern struct ext2_group_desc *ext2fs_group_desc(ext2_filsys fs,
struct opaque_ext2_group_desc *gdp,
dgrp_t group);
extern blk64_t ext2fs_block_bitmap_csum(ext2_filsys fs, dgrp_t group);
extern blk64_t ext2fs_block_bitmap_loc(ext2_filsys fs, dgrp_t group);
extern void ext2fs_block_bitmap_loc_set(ext2_filsys fs, dgrp_t group,
blk64_t blk);
extern __u32 ext2fs_inode_bitmap_csum(ext2_filsys fs, dgrp_t group);
extern blk64_t ext2fs_inode_bitmap_loc(ext2_filsys fs, dgrp_t group);
extern void ext2fs_inode_bitmap_loc_set(ext2_filsys fs, dgrp_t group,
blk64_t blk);
extern blk64_t ext2fs_inode_table_loc(ext2_filsys fs, dgrp_t group);
extern void ext2fs_inode_table_loc_set(ext2_filsys fs, dgrp_t group,
blk64_t blk);
extern __u32 ext2fs_bg_free_blocks_count(ext2_filsys fs, dgrp_t group);
extern void ext2fs_bg_free_blocks_count_set(ext2_filsys fs, dgrp_t group,
__u32 n);
extern __u32 ext2fs_bg_free_inodes_count(ext2_filsys fs, dgrp_t group);

```

```

extern void ext2fs_bg_free_inodes_count_set(ext2_filsys fs, dgrp_t group,
__u32 n);
extern __u32 ext2fs_bg_used_dirs_count(ext2_filsys fs, dgrp_t group);
extern void ext2fs_bg_used_dirs_count_set(ext2_filsys fs, dgrp_t group,
__u32 n);
extern __u32 ext2fs_bg_itable_unused(ext2_filsys fs, dgrp_t group);
extern void ext2fs_bg_itable_unused_set(ext2_filsys fs, dgrp_t group,
__u32 n);
extern __u16 ext2fs_bg_flags(ext2_filsys fs, dgrp_t group);
extern void ext2fs_bg_flags_zap(ext2_filsys fs, dgrp_t group);
extern int ext2fs_bg_flags_test(ext2_filsys fs, dgrp_t group, __u16 bg_flag);
extern void ext2fs_bg_flags_set(ext2_filsys fs, dgrp_t group, __u16 bg_flags);
extern void ext2fs_bg_flags_clear(ext2_filsys fs, dgrp_t group, __u16 bg_flags);
extern __u16 ext2fs_bg_checksum(ext2_filsys fs, dgrp_t group);
extern void ext2fs_bg_checksum_set(ext2_filsys fs, dgrp_t group, __u16 checksum);
extern blk64_t ext2fs_file_acl_block(ext2_filsys fs,
const struct ext2_inode *inode);
extern void ext2fs_file_acl_block_set(ext2_filsys fs,
struct ext2_inode *inode, blk64_t blk);
extern errcode_t ext2fs_inode_size_set(ext2_filsys fs, struct ext2_inode *inode,
ext2_off64_t size);

/* block.c */
extern errcode_t ext2fs_block_iterate(ext2_filsys fs,
ext2_ino_tino,
intflags,
char *block_buf,
int (*func)(ext2_filsys fs,
blk_t*blocknr,
intblockcnt,
void*priv_data),
void *priv_data);
errcode_t ext2fs_block_iterate2(ext2_filsys fs,
ext2_ino_tino,
intflags,
char *block_buf,
int (*func)(ext2_filsys fs,
blk_t*blocknr,
e2_blkcnt_tblockcnt,
blk_tref_blk,
intref_offset,
void*priv_data),
void *priv_data);
errcode_t ext2fs_block_iterate3(ext2_filsys fs,
ext2_ino_tino,
intflags,
char *block_buf,
int (*func)(ext2_filsys fs,
blk64_t*blocknr,
e2_blkcnt_tblockcnt,
blk64_tref_blk,
intref_offset,
void*priv_data),
void *priv_data);

```

```

/* bmap.c */
extern errcode_t ext2fs_bmap(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode,
    char *block_buf, int bmap_flags,
    blk_t block, blk_t *phys_blk);
extern errcode_t ext2fs_bmap2(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode,
    char *block_buf, int bmap_flags, blk64_t block,
    int *ret_flags, blk64_t *phys_blk);
errcode_t ext2fs_map_cluster_block(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode, blk64_t lblk,
    blk64_t *pblk);

#if 0
/* bmove.c */
extern errcode_t ext2fs_move_blocks(ext2_filsys fs,
    ext2fs_block_bitmap reserve,
    ext2fs_block_bitmap alloc_map,
    int flags);
#endif

/* check_desc.c */
extern errcode_t ext2fs_check_desc(ext2_filsys fs);

/* closefs.c */
extern errcode_t ext2fs_close(ext2_filsys fs);
extern errcode_t ext2fs_close2(ext2_filsys fs, int flags);
extern errcode_t ext2fs_close_free(ext2_filsys *fs);
extern errcode_t ext2fs_flush(ext2_filsys fs);
extern errcode_t ext2fs_flush2(ext2_filsys fs, int flags);
extern int ext2fs_bg_has_super(ext2_filsys fs, dgrp_t group_block);
extern errcode_t ext2fs_super_and_bgd_loc2(ext2_filsys fs,
    dgrp_t group,
    blk64_t *ret_super_blk,
    blk64_t *ret_old_desc_blk,
    blk64_t *ret_new_desc_blk,
    blk_t *ret_used_blks);
extern int ext2fs_super_and_bgd_loc(ext2_filsys fs,
    dgrp_t group,
    blk_t *ret_super_blk,
    blk_t *ret_old_desc_blk,
    blk_t *ret_new_desc_blk,
    int *ret_meta_bg);
extern void ext2fs_update_dynamic_rev(ext2_filsys fs);

/* crc32c.c */
extern __u32 ext2fs_crc32_be(__u32 crc, unsigned char const *p, size_t len);
extern __u32 ext2fs_crc32c_le(__u32 crc, unsigned char const *p, size_t len);

/* csum.c */
extern void ext2fs_init_csum_seed(ext2_filsys fs);
extern errcode_t ext2fs_mmp_csum_set(ext2_filsys fs, struct mmp_struct *mmp);
extern int ext2fs_mmp_csum_verify(ext2_filsys fs, struct mmp_struct *mmp);
extern int ext2fs_verify_csum_type(ext2_filsys fs, struct ext2_super_block *sb);

```

```

extern errcode_t ext2fs_superblock_csum_set(ext2_filsys fs,
    struct ext2_super_block *sb);
extern int ext2fs_superblock_csum_verify(ext2_filsys fs,
    struct ext2_super_block *sb);
extern errcode_t ext2fs_ext_attr_block_csum_set(ext2_filsys fs,
    ext2_ino_t inum, blk64_t block,
    struct ext2_ext_attr_header *hdr);
extern int ext2fs_ext_attr_block_csum_verify(ext2_filsys fs, ext2_ino_t inum,
    blk64_t block,
    struct ext2_ext_attr_header *hdr);
#define EXT2_DIRENT_TAIL(block, blocksize) \
(((struct ext2_dir_entry_tail *)(((char *) (block)) + \
(blocksize) - sizeof(struct ext2_dir_entry_tail)))

extern void ext2fs_initialize_dirent_tail(ext2_filsys fs,
    struct ext2_dir_entry_tail *t);
extern int ext2fs_dirent_has_tail(ext2_filsys fs,
    struct ext2_dir_entry *dirent);
extern int ext2fs_dirent_csum_verify(ext2_filsys fs, ext2_ino_t inum,
    struct ext2_dir_entry *dirent);
extern int ext2fs_dir_block_csum_verify(ext2_filsys fs, ext2_ino_t inum,
    struct ext2_dir_entry *dirent);
extern errcode_t ext2fs_dir_block_csum_set(ext2_filsys fs, ext2_ino_t inum,
    struct ext2_dir_entry *dirent);
extern errcode_t ext2fs_get_dx_countlimit(ext2_filsys fs,
    struct ext2_dir_entry *dirent,
    struct ext2_dx_countlimit **cc,
    int *offset);
extern errcode_t ext2fs_extent_block_csum_set(ext2_filsys fs,
    ext2_ino_t inum,
    struct ext3_extent_header *eh);
extern int ext2fs_extent_block_csum_verify(ext2_filsys fs,
    ext2_ino_t inum,
    struct ext3_extent_header *eh);
extern errcode_t ext2fs_block_bitmap_csum_set(ext2_filsys fs, dgrp_t group,
    char *bitmap, int size);
extern int ext2fs_block_bitmap_csum_verify(ext2_filsys fs, dgrp_t group,
    char *bitmap, int size);
extern errcode_t ext2fs_inode_bitmap_csum_set(ext2_filsys fs, dgrp_t group,
    char *bitmap, int size);
extern int ext2fs_inode_bitmap_csum_verify(ext2_filsys fs, dgrp_t group,
    char *bitmap, int size);
extern errcode_t ext2fs_inode_csum_set(ext2_filsys fs, ext2_ino_t inum,
    struct ext2_inode_large *inode);
extern int ext2fs_inode_csum_verify(ext2_filsys fs, ext2_ino_t inum,
    struct ext2_inode_large *inode);
extern void ext2fs_group_desc_csum_set(ext2_filsys fs, dgrp_t group);
extern int ext2fs_group_desc_csum_verify(ext2_filsys fs, dgrp_t group);
extern errcode_t ext2fs_set_gdt_csum(ext2_filsys fs);
extern __u16 ext2fs_group_desc_csum(ext2_filsys fs, dgrp_t group);

/* dblist.c */
extern errcode_t ext2fs_init_dblist(ext2_filsys fs, ext2_dblist *ret_dblist);
extern errcode_t ext2fs_add_dir_block(ext2_dblist dblist, ext2_ino_t ino,

```

```

    blk_t blk, int blockcnt);
extern errcode_t ext2fs_add_dir_block2(ext2_dblist dblist, ext2_ino_t ino,
    blk64_t blk, e2_blkcnt_t blockcnt);
extern void ext2fs_dblist_sort(ext2_dblist dblist,
    EXT2_QSORT_TYPE (*sortfunc)(const void *,
    const void *));
extern void ext2fs_dblist_sort2(ext2_dblist dblist,
    EXT2_QSORT_TYPE (*sortfunc)(const void *,
    const void *));
extern errcode_t ext2fs_dblist_iterate(ext2_dblist dblist,
    int (*func)(ext2_filsys fs, struct ext2_db_entry *db_info,
    void*priv_data),
    void *priv_data);
extern errcode_t ext2fs_dblist_iterate2(ext2_dblist dblist,
    int (*func)(ext2_filsys fs, struct ext2_db_entry2 *db_info,
    void*priv_data),
    void *priv_data);
extern errcode_t ext2fs_dblist_iterate3(ext2_dblist dblist,
    int (*func)(ext2_filsys fs, struct ext2_db_entry2 *db_info,
    void*priv_data),
    unsigned long long start,
    unsigned long long count,
    void *priv_data);
extern errcode_t ext2fs_set_dir_block(ext2_dblist dblist, ext2_ino_t ino,
    blk_t blk, int blockcnt);
extern errcode_t ext2fs_set_dir_block2(ext2_dblist dblist, ext2_ino_t ino,
    blk64_t blk, e2_blkcnt_t blockcnt);
extern errcode_t ext2fs_copy_dblist(ext2_dblist src,
    ext2_dblist *dest);
extern int ext2fs_dblist_count(ext2_dblist dblist);
extern blk64_t ext2fs_dblist_count2(ext2_dblist dblist);
extern errcode_t ext2fs_dblist_get_last(ext2_dblist dblist,
    struct ext2_db_entry **entry);
extern errcode_t ext2fs_dblist_get_last2(ext2_dblist dblist,
    struct ext2_db_entry2 **entry);
extern errcode_t ext2fs_dblist_drop_last(ext2_dblist dblist);

/* dblist_dir.c */
extern errcode_t
ext2fs_dblist_dir_iterate(ext2_dblist dblist,
    intflags,
    char*block_buf,
    int (*func)(ext2_ino_tdir,
    intentry,
    struct ext2_dir_entry *dirent,
    intoffset,
    intblocksize,
    char*buf,
    void*priv_data),
    void *priv_data);

#if 0
/* digest_encode.c */
#define EXT2FS_DIGEST_SIZE EXT2FS_SHA256_LENGTH
extern int ext2fs_digest_encode(const char *src, int len, char *dst);

```

```
extern int ext2fs_digest_decode(const char *src, int len, char *dst);
#endif

/* dirblock.c */
extern errcode_t ext2fs_read_dir_block(ext2_filsys fs, blk_t block,
    void *buf);
extern errcode_t ext2fs_read_dir_block2(ext2_filsys fs, blk_t block,
    void *buf, int flags);
extern errcode_t ext2fs_read_dir_block3(ext2_filsys fs, blk64_t block,
    void *buf, int flags);
extern errcode_t ext2fs_read_dir_block4(ext2_filsys fs, blk64_t block,
    void *buf, int flags, ext2_ino_t ino);
extern errcode_t ext2fs_write_dir_block(ext2_filsys fs, blk_t block,
    void *buf);
extern errcode_t ext2fs_write_dir_block2(ext2_filsys fs, blk_t block,
    void *buf, int flags);
extern errcode_t ext2fs_write_dir_block3(ext2_filsys fs, blk64_t block,
    void *buf, int flags);
extern errcode_t ext2fs_write_dir_block4(ext2_filsys fs, blk64_t block,
    void *buf, int flags, ext2_ino_t ino);

/* dirhash.c */
extern errcode_t ext2fs_dirhash(int version, const char *name, int len,
    const __u32 *seed,
    ext2_dirhash_t *ret_hash,
    ext2_dirhash_t *ret_minor_hash);

/* dir_iterate.c */
extern errcode_t ext2fs_get_rec_len(ext2_filsys fs,
    struct ext2_dir_entry *dirent,
    unsigned int *rec_len);
extern errcode_t ext2fs_set_rec_len(ext2_filsys fs,
    unsigned int len,
    struct ext2_dir_entry *dirent);
extern errcode_t ext2fs_dir_iterate(ext2_filsys fs,
    ext2_ino_t dir,
    int flags,
    char *block_buf,
    int (*func)(struct ext2_dir_entry *dirent,
    intoffset,
    intblocksize,
    char*buf,
    void*priv_data),
    void *priv_data);
extern errcode_t ext2fs_dir_iterate2(ext2_filsys fs,
    ext2_ino_t dir,
    int flags,
    char *block_buf,
    int (*func)(ext2_ino_tdir,
    intentry,
    struct ext2_dir_entry *dirent,
    intoffset,
    intblocksize,
```

```

char*buf,
void*priv_data),
void *priv_data);

/* dupfs.c */
extern errcode_t ext2fs_dup_handle(ext2_filsys src, ext2_filsys *dest);

/* expanddir.c */
extern errcode_t ext2fs_expand_dir(ext2_filsys fs, ext2_ino_t dir);

/* ext_attr.c */
extern __u32 ext2fs_ext_attr_hash_entry(struct ext2_ext_attr_entry *entry,
void *data);
extern errcode_t ext2fs_read_ext_attr(ext2_filsys fs, blk_t block, void *buf);
extern errcode_t ext2fs_read_ext_attr2(ext2_filsys fs, blk64_t block,
void *buf);
extern errcode_t ext2fs_read_ext_attr3(ext2_filsys fs, blk64_t block,
void *buf, ext2_ino_t inum);
extern errcode_t ext2fs_write_ext_attr(ext2_filsys fs, blk_t block,
void *buf);
extern errcode_t ext2fs_write_ext_attr2(ext2_filsys fs, blk64_t block,
void *buf);
extern errcode_t ext2fs_write_ext_attr3(ext2_filsys fs, blk64_t block,
void *buf, ext2_ino_t inum);
extern errcode_t ext2fs_adjust_ea_refcount(ext2_filsys fs, blk_t blk,
char *block_buf,
int adjust, __u32 *newcount);
extern errcode_t ext2fs_adjust_ea_refcount2(ext2_filsys fs, blk64_t blk,
char *block_buf,
int adjust, __u32 *newcount);
extern errcode_t ext2fs_adjust_ea_refcount3(ext2_filsys fs, blk64_t blk,
char *block_buf,
int adjust, __u32 *newcount,
ext2_ino_t inum);
errcode_t ext2fs_xattrs_write(struct ext2_xattr_handle *handle);
errcode_t ext2fs_xattrs_read(struct ext2_xattr_handle *handle);
errcode_t ext2fs_xattrs_iterate(struct ext2_xattr_handle *h,
int (*func)(char *name, char *value,
size_t value_len, void *data),
void *data);
errcode_t ext2fs_xattr_get(struct ext2_xattr_handle *h, const char *key,
void **value, size_t *value_len);
errcode_t ext2fs_xattr_set(struct ext2_xattr_handle *handle,
const char *key,
const void *value,
size_t value_len);
errcode_t ext2fs_xattr_remove(struct ext2_xattr_handle *handle,
const char *key);
errcode_t ext2fs_xattrs_open(ext2_filsys fs, ext2_ino_t ino,
struct ext2_xattr_handle **handle);
errcode_t ext2fs_xattrs_close(struct ext2_xattr_handle **handle);
errcode_t ext2fs_free_ext_attr(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode_large *inode);
errcode_t ext2fs_xattrs_count(struct ext2_xattr_handle *handle, size_t *count);
errcode_t ext2fs_xattr_inode_max_size(ext2_filsys fs, ext2_ino_t ino,

```



```

    size_t *size);

/* extent.c */
extern errcode_t ext2fs_extent_header_verify(void *ptr, int size);
extern errcode_t ext2fs_extent_open(ext2_filsys fs, ext2_ino_t ino,
    ext2_extent_handle_t *handle);
extern errcode_t ext2fs_extent_open2(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode,
    ext2_extent_handle_t *ret_handle);
extern void ext2fs_extent_free(ext2_extent_handle_t handle);
extern errcode_t ext2fs_extent_get(ext2_extent_handle_t handle,
    int flags, struct ext2fs_extent *extent);
extern errcode_t ext2fs_extent_node_split(ext2_extent_handle_t handle);
extern errcode_t ext2fs_extent_replace(ext2_extent_handle_t handle, int flags,
    struct ext2fs_extent *extent);
extern errcode_t ext2fs_extent_insert(ext2_extent_handle_t handle, int flags,
    struct ext2fs_extent *extent);
extern errcode_t ext2fs_extent_set_bmap(ext2_extent_handle_t handle,
    blk64_t logical, blk64_t physical,
    int flags);
extern errcode_t ext2fs_extent_delete(ext2_extent_handle_t handle, int flags);
extern errcode_t ext2fs_extent_get_info(ext2_extent_handle_t handle,
    struct ext2_extent_info *info);
extern errcode_t ext2fs_extent_goto(ext2_extent_handle_t handle,
    blk64_t blk);
extern errcode_t ext2fs_extent_goto2(ext2_extent_handle_t handle,
    int leaf_level, blk64_t blk);
extern errcode_t ext2fs_extent_fix_parents(ext2_extent_handle_t handle);
size_t ext2fs_max_extent_depth(ext2_extent_handle_t handle);

/* fallocate.c */
#define EXT2_FALLOCATE_ZERO_BLOCKS(0x1)
#define EXT2_FALLOCATE_FORCE_INIT(0x2)
#define EXT2_FALLOCATE_FORCE_UNINIT(0x4)
#define EXT2_FALLOCATE_INIT_BEYOND_EOF(0x8)
#define EXT2_FALLOCATE_ALL_FLAGS(0xF)
errcode_t ext2fs_fallocate(ext2_filsys fs, int flags, ext2_ino_t ino,
    struct ext2_inode *inode, blk64_t goal,
    blk64_t start, blk64_t len);

/* fileio.c */
extern errcode_t ext2fs_file_open2(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode,
    int flags, ext2_file_t *ret);
extern errcode_t ext2fs_file_open(ext2_filsys fs, ext2_ino_t ino,
    int flags, ext2_file_t *ret);
extern ext2_filsys ext2fs_file_get_fs(ext2_file_t file);
struct ext2_inode *ext2fs_file_get_inode(ext2_file_t file);
extern ext2_ino_t ext2fs_file_get_inode_num(ext2_file_t file);
extern errcode_t ext2fs_file_close(ext2_file_t file);
extern errcode_t ext2fs_file_flush(ext2_file_t file);
extern errcode_t ext2fs_file_read(ext2_file_t file, void *buf,
    unsigned int wanted, unsigned int *got);
extern errcode_t ext2fs_file_write(ext2_file_t file, const void *buf,

```

```

    unsigned int nbytes, unsigned int *written);
extern errcode_t ext2fs_file_llseek(ext2_file_t file, __u64 offset,
    int whence, __u64 *ret_pos);
extern errcode_t ext2fs_file_lseek(ext2_file_t file, ext2_off_t offset,
    int whence, ext2_off_t *ret_pos);
errcode_t ext2fs_file_get_lsize(ext2_file_t file, __u64 *ret_size);
extern ext2_off_t ext2fs_file_get_size(ext2_file_t file);
extern errcode_t ext2fs_file_set_size(ext2_file_t file, ext2_off_t size);
extern errcode_t ext2fs_file_set_size2(ext2_file_t file, ext2_off64_t size);

/* finddev.c */
extern char *ext2fs_find_block_device(dev_t device);

/* flushb.c */
extern errcode_t ext2fs_sync_device(int fd, int flushb);

/* freefs.c */
extern void ext2fs_free(ext2_filsys fs);
extern void ext2fs_free_dbllist(ext2_dbllist dblist);
extern void ext2fs_badblocks_list_free(ext2_badblocks_list bb);
extern void ext2fs_u32_list_free(ext2_u32_list bb);

/* gen_bitmap.c */
extern void ext2fs_free_generic_bitmap(ext2fs_inode_bitmap bitmap);
extern errcode_t ext2fs_make_generic_bitmap(errcode_t magic, ext2_filsys fs,
    __u32 start, __u32 end,
    __u32 real_end,
    const char *descr, char *init_map,
    ext2fs_generic_bitmap *ret);
extern errcode_t ext2fs_allocate_generic_bitmap(__u32 start,
    __u32 end,
    __u32 real_end,
    const char *descr,
    ext2fs_generic_bitmap *ret);
extern errcode_t ext2fs_copy_generic_bitmap(ext2fs_generic_bitmap src,
    ext2fs_generic_bitmap *dest);
extern void ext2fs_clear_generic_bitmap(ext2fs_generic_bitmap bitmap);
extern errcode_t ext2fs_fudge_generic_bitmap_end(ext2fs_inode_bitmap bitmap,
    errcode_t magic,
    errcode_t neq,
    ext2_ino_t end,
    ext2_ino_t *oend);
extern void ext2fs_set_generic_bitmap_padding(ext2fs_generic_bitmap map);
extern errcode_t ext2fs_resize_generic_bitmap(errcode_t magic,
    __u32 new_end,
    __u32 new_real_end,
    ext2fs_generic_bitmap bmap);
extern errcode_t ext2fs_compare_generic_bitmap(errcode_t magic, errcode_t neq,
    ext2fs_generic_bitmap bm1,
    ext2fs_generic_bitmap bm2);
extern errcode_t ext2fs_get_generic_bitmap_range(ext2fs_generic_bitmap bmap,
    errcode_t magic,
    __u32 start, __u32 num,
    void *out);
extern errcode_t ext2fs_set_generic_bitmap_range(ext2fs_generic_bitmap bmap,

```

```

    errcode_t magic,
    __u32 start, __u32 num,
    void *in);
extern errcode_t ext2fs_find_first_zero_generic_bitmap(ext2fs_generic_bitmap bitmap,
    __u32 start, __u32 end,
    __u32 *out);
extern errcode_t ext2fs_find_first_set_generic_bitmap(ext2fs_generic_bitmap bitmap,
    __u32 start, __u32 end,
    __u32 *out);

/* gen_bitmap64.c */
void ext2fs_free_generic_bmap(ext2fs_generic_bitmap bmap);
errcode_t ext2fs_alloc_generic_bmap(ext2_filsys fs, errcode_t magic,
    int type, __u64 start, __u64 end,
    __u64 real_end,
    const char *descr,
    ext2fs_generic_bitmap *ret);
errcode_t ext2fs_copy_generic_bmap(ext2fs_generic_bitmap src,
    ext2fs_generic_bitmap *dest);
void ext2fs_clear_generic_bmap(ext2fs_generic_bitmap bitmap);
errcode_t ext2fs_fudge_generic_bmap_end(ext2fs_generic_bitmap bitmap,
    errcode_t neq,
    __u64 end, __u64 *oend);
void ext2fs_set_generic_bmap_padding(ext2fs_generic_bitmap bmap);
errcode_t ext2fs_resize_generic_bmap(ext2fs_generic_bitmap bmap,
    __u64 new_end,
    __u64 new_real_end);
errcode_t ext2fs_compare_generic_bmap(errcode_t neq,
    ext2fs_generic_bitmap bm1,
    ext2fs_generic_bitmap bm2);
errcode_t ext2fs_get_generic_bmap_range(ext2fs_generic_bitmap bmap,
    __u64 start, unsigned int num,
    void *out);
errcode_t ext2fs_set_generic_bmap_range(ext2fs_generic_bitmap bmap,
    __u64 start, unsigned int num,
    void *in);
errcode_t ext2fs_convert_subcluster_bitmap(ext2_filsys fs,
    ext2fs_block_bitmap *bitmap);

/* get_num_dirs.c */
extern errcode_t ext2fs_get_num_dirs(ext2_filsys fs, ext2_ino_t *ret_num_dirs);

/* getsize.c */
extern errcode_t ext2fs_get_device_size(const char *file, int blocksize,
    blk_t *retblocks);
extern errcode_t ext2fs_get_device_size2(const char *file, int blocksize,
    blk64_t *retblocks);

/* getsectsize.c */
extern int ext2fs_get_dio_alignment(int fd);
errcode_t ext2fs_get_device_sectsize(const char *file, int *sectsize);
errcode_t ext2fs_get_device_phys_sectsize(const char *file, int *sectsize);

/* i_block.c */

```

```

errcode_t ext2fs_iblk_add_blocks(ext2_filsys fs, struct ext2_inode *inode,
    blk64_t num_blocks);
errcode_t ext2fs_iblk_sub_blocks(ext2_filsys fs, struct ext2_inode *inode,
    blk64_t num_blocks);
errcode_t ext2fs_iblk_set(ext2_filsys fs, struct ext2_inode *inode, blk64_t b);

/* imager.c */
extern errcode_t ext2fs_image_inode_write(ext2_filsys fs, int fd, int flags);
extern errcode_t ext2fs_image_inode_read(ext2_filsys fs, int fd, int flags);
extern errcode_t ext2fs_image_super_write(ext2_filsys fs, int fd, int flags);
extern errcode_t ext2fs_image_super_read(ext2_filsys fs, int fd, int flags);
extern errcode_t ext2fs_image_bitmap_write(ext2_filsys fs, int fd, int flags);
extern errcode_t ext2fs_image_bitmap_read(ext2_filsys fs, int fd, int flags);

/* ind_block.c */
errcode_t ext2fs_read_ind_block(ext2_filsys fs, blk_t blk, void *buf);
errcode_t ext2fs_write_ind_block(ext2_filsys fs, blk_t blk, void *buf);

/* initialize.c */
extern errcode_t ext2fs_initialize(const char *name, int flags,
    struct ext2_super_block *param,
    io_manager manager, ext2_filsys *ret_fs);

/* icount.c */
extern void ext2fs_free_icount(ext2_icount_t icount);
extern errcode_t ext2fs_create_icount_tdb(ext2_filsys fs, char *tdb_dir,
    int flags, ext2_icount_t *ret);
extern errcode_t ext2fs_create_icount2(ext2_filsys fs, int flags,
    unsigned int size,
    ext2_icount_t hint, ext2_icount_t *ret);
extern errcode_t ext2fs_create_icount(ext2_filsys fs, int flags,
    unsigned int size,
    ext2_icount_t *ret);
extern errcode_t ext2fs_icount_fetch(ext2_icount_t icount, ext2_ino_t ino,
    __u16 *ret);
extern errcode_t ext2fs_icount_increment(ext2_icount_t icount, ext2_ino_t ino,
    __u16 *ret);
extern errcode_t ext2fs_icount_decrement(ext2_icount_t icount, ext2_ino_t ino,
    __u16 *ret);
extern errcode_t ext2fs_icount_store(ext2_icount_t icount, ext2_ino_t ino,
    __u16 count);
extern ext2_ino_t ext2fs_get_icount_size(ext2_icount_t icount);
errcode_t ext2fs_icount_validate(ext2_icount_t icount, FILE *);

/* inline.c */

extern errcode_t ext2fs_get_memalign(unsigned long size,
    unsigned long align, void *ptr);

/* inline_data.c */
extern errcode_t ext2fs_inline_data_init(ext2_filsys fs, ext2_ino_t ino);
extern errcode_t ext2fs_inline_data_size(ext2_filsys fs, ext2_ino_t ino,
    size_t *size);
extern errcode_t ext2fs_inline_data_get(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode *inode,

```

```

void *buf, size_t *size);
extern errcode_t ext2fs_inline_data_set(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode *inode,
void *buf, size_t size);

/* inode.c */
extern errcode_t ext2fs_create_inode_cache(ext2_filsys fs,
    unsigned int cache_size);
extern void ext2fs_free_inode_cache(struct ext2_inode_cache *icache);
extern errcode_t ext2fs_flush_icache(ext2_filsys fs);
extern errcode_t ext2fs_get_next_inode_full(ext2_inode_scan scan,
    ext2_ino_t *ino,
    struct ext2_inode *inode,
    int bufsize);
#define EXT2_INODE_SCAN_DEFAULT_BUFFER_BLOCKS8
extern errcode_t ext2fs_open_inode_scan(ext2_filsys fs, int buffer_blocks,
    ext2_inode_scan *ret_scan);
extern void ext2fs_close_inode_scan(ext2_inode_scan scan);
extern errcode_t ext2fs_get_next_inode(ext2_inode_scan scan, ext2_ino_t *ino,
    struct ext2_inode *inode);
extern errcode_t ext2fs_inode_scan_goto_blockgroup(ext2_inode_scan scan,
    intgroup);
extern void ext2fs_set_inode_callback
(ext2_inode_scan scan,
errcode_t (*done_group)(ext2_filsys fs,
ext2_inode_scan scan,
dgrp_t group,
void *priv_data),
void *done_group_data);
extern int ext2fs_inode_scan_flags(ext2_inode_scan scan, int set_flags,
    int clear_flags);
extern errcode_t ext2fs_read_inode_full(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode * inode,
int bufsize);
extern errcode_t ext2fs_read_inode (ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode * inode);
extern errcode_t ext2fs_write_inode_full(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode * inode,
int bufsize);
extern errcode_t ext2fs_write_inode(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode * inode);
extern errcode_t ext2fs_write_new_inode(ext2_filsys fs, ext2_ino_t ino,
    struct ext2_inode * inode);
extern errcode_t ext2fs_get_blocks(ext2_filsys fs, ext2_ino_t ino, blk_t *blocks);
extern errcode_t ext2fs_check_directory(ext2_filsys fs, ext2_ino_t ino);

/* inode_io.c */
extern io_manager inode_io_manager;
extern errcode_t ext2fs_inode_io_intern(ext2_filsys fs, ext2_ino_t ino,
char **name);
extern errcode_t ext2fs_inode_io_intern2(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode *inode,
char **name);

```

```

/* ismounted.c */
extern errcode_t ext2fs_check_if_mounted(const char *file, int *mount_flags);
extern errcode_t ext2fs_check_mount_point(const char *device, int *mount_flags,
char *mtpnt, int mtlent);

/* punch.c */
/*
 * NOTE: This function removes from an inode the blocks "start", "end", and
 * every block in between.
 */
extern errcode_t ext2fs_punch(ext2_filsys fs, ext2_ino_t ino,
struct ext2_inode *inode,
char *block_buf, blk64_t start,
blk64_t end);

/* namei.c */
extern errcode_t ext2fs_lookup(ext2_filsys fs, ext2_ino_t dir, const char *name,
int namelen, char *buf, ext2_ino_t *inode);
extern errcode_t ext2fs_namei(ext2_filsys fs, ext2_ino_t root, ext2_ino_t cwd,
const char *name, ext2_ino_t *inode);
extern errcode_t ext2fs_namei_follow(ext2_filsys fs, ext2_ino_t root, ext2_ino_t cwd,
const char *name, ext2_ino_t *inode);
extern errcode_t ext2fs_follow_link(ext2_filsys fs, ext2_ino_t root, ext2_ino_t cwd,
ext2_ino_t inode, ext2_ino_t *res_inode);

/* native.c */
int ext2fs_native_flag(void);

/* newdir.c */
extern errcode_t ext2fs_new_dir_block(ext2_filsys fs, ext2_ino_t dir_ino,
ext2_ino_t parent_ino, char **block);
extern errcode_t ext2fs_new_dir_inline_data(ext2_filsys fs, ext2_ino_t dir_ino,
ext2_ino_t parent_ino, __u32 *iblock);

/* mkdir.c */
extern errcode_t ext2fs_mkdir(ext2_filsys fs, ext2_ino_t parent, ext2_ino_t inum,
const char *name);

/* mkjournal.c */
extern errcode_t ext2fs_zero_blocks(ext2_filsys fs, blk_t blk, int num,
blk_t *ret_blk, int *ret_count);
extern errcode_t ext2fs_zero_blocks2(ext2_filsys fs, blk64_t blk, int num,
blk64_t *ret_blk, int *ret_count);
extern errcode_t ext2fs_create_journal_superblock(ext2_filsys fs,
__u32 num_blocks, int flags,
char **ret_jsb);
extern errcode_t ext2fs_add_journal_device(ext2_filsys fs,
ext2_filsys journal_dev);
extern errcode_t ext2fs_add_journal_inode(ext2_filsys fs, blk_t num_blocks,
int flags);
extern errcode_t ext2fs_add_journal_inode2(ext2_filsys fs, blk_t num_blocks,
blk64_t goal, int flags);
extern int ext2fs_default_journal_size(__u64 num_blocks);
extern int ext2fs_journal_sb_start(int blocksize);

```

```

/* openfs.c */
extern errcode_t ext2fs_open(const char *name, int flags, int superblock,
    unsigned int block_size, io_manager manager,
    ext2_filsys *ret_fs);
extern errcode_t ext2fs_open2(const char *name, const char *io_options,
    int flags, int superblock,
    unsigned int block_size, io_manager manager,
    ext2_filsys *ret_fs);
/*
 * The dgrp_t argument to these two functions is not actually a group number
 * but a block number offset within a group table! Convert with the formula
 * (group_number / groups_per_block).
 */
extern blk64_t ext2fs_descriptor_block_loc2(ext2_filsys fs,
    blk64_t group_block, dgrp_t i);
extern blk_t ext2fs_descriptor_block_loc(ext2_filsys fs, blk_t group_block,
    dgrp_t i);
errcode_t ext2fs_get_data_io(ext2_filsys fs, io_channel *old_io);
errcode_t ext2fs_set_data_io(ext2_filsys fs, io_channel new_io);
errcode_t ext2fs_rewrite_to_io(ext2_filsys fs, io_channel new_io);

/* get_pathname.c */
extern errcode_t ext2fs_get_pathname(ext2_filsys fs, ext2_ino_t dir, ext2_ino_t ino,
    char **name);

/* link.c */
errcode_t ext2fs_link(ext2_filsys fs, ext2_ino_t dir, const char *name,
    ext2_ino_t ino, int flags);
errcode_t ext2fs_unlink(ext2_filsys fs, ext2_ino_t dir, const char *name,
    ext2_ino_t ino, int flags);

/* symlink.c */
errcode_t ext2fs_symlink(ext2_filsys fs, ext2_ino_t parent, ext2_ino_t ino,
    const char *name, const char *target);

/* mmp.c */
errcode_t ext2fs_mmp_read(ext2_filsys fs, blk64_t mmp_blk, void *buf);
errcode_t ext2fs_mmp_write(ext2_filsys fs, blk64_t mmp_blk, void *buf);
errcode_t ext2fs_mmp_clear(ext2_filsys fs);
errcode_t ext2fs_mmp_init(ext2_filsys fs);
errcode_t ext2fs_mmp_start(ext2_filsys fs);
errcode_t ext2fs_mmp_update(ext2_filsys fs);
errcode_t ext2fs_mmp_update2(ext2_filsys fs, int immediately);
errcode_t ext2fs_mmp_stop(ext2_filsys fs);
unsigned ext2fs_mmp_new_seq(void);

/* read_bb.c */
extern errcode_t ext2fs_read_bb_inode(ext2_filsys fs,
    ext2_badblocks_list *bb_list);

/* read_bb_file.c */
extern errcode_t ext2fs_read_bb_FILE2(ext2_filsys fs, FILE *f,
    ext2_badblocks_list *bb_list,
    void *priv_data,

```

```

    void (*invalid)(ext2_filsys fs,
        blk_t blk,
        char *badstr,
        void *priv_data));
extern errcode_t ext2fs_read_bb_FILE(ext2_filsys fs, FILE *f,
    ext2_badblocks_list *bb_list,
    void (*invalid)(ext2_filsys fs,
        blk_t blk));

/* res_gdt.c */
extern errcode_t ext2fs_create_resize_inode(ext2_filsys fs);

/* sha256.c */
#define EXT2FS_SHA256_LENGTH 32
#if 0
extern void ext2fs_sha256(const unsigned char *in, unsigned long in_size,
    unsigned char out[EXT2FS_SHA256_LENGTH]);
#endif

/* sha512.c */
#define EXT2FS_SHA512_LENGTH 64
extern void ext2fs_sha512(const unsigned char *in, unsigned long in_size,
    unsigned char out[EXT2FS_SHA512_LENGTH]);

/* swapfs.c */
extern errcode_t ext2fs_dirent_swab_in2(ext2_filsys fs, char *buf, size_t size,
    int flags);
extern errcode_t ext2fs_dirent_swab_in(ext2_filsys fs, char *buf, int flags);
extern errcode_t ext2fs_dirent_swab_out2(ext2_filsys fs, char *buf, size_t size,
    int flags);
extern errcode_t ext2fs_dirent_swab_out(ext2_filsys fs, char *buf, int flags);
extern void ext2fs_swap_ext_attr(char *to, char *from, int bufsize,
    int has_header);
extern void ext2fs_swap_ext_attr_header(struct ext2_ext_attr_header *to_header,
    struct ext2_ext_attr_header *from_hdr);
extern void ext2fs_swap_ext_attr_entry(struct ext2_ext_attr_entry *to_entry,
    struct ext2_ext_attr_entry *from_entry);
extern void ext2fs_swap_super(struct ext2_super_block *super);
extern void ext2fs_swap_group_desc(struct ext2_group_desc *gdp);
extern void ext2fs_swap_group_desc2(ext2_filsys, struct ext2_group_desc *gdp);
extern void ext2fs_swap_inode_full(ext2_filsys fs, struct ext2_inode_large *t,
    struct ext2_inode_large *f, int hostorder,
    int bufsize);
extern void ext2fs_swap_inode(ext2_filsys fs, struct ext2_inode *t,
    struct ext2_inode *f, int hostorder);
extern void ext2fs_swap_mmp(struct mmp_struct *mmp);

/* unix_io.c */
extern int ext2fs_open_file(const char *pathname, int flags, mode_t mode);
extern int ext2fs_stat(const char *path, ext2fs_struct_stat *buf);
extern int ext2fs_fstat(int fd, ext2fs_struct_stat *buf);

/* valid_blk.c */
extern int ext2fs_inode_has_valid_blocks(struct ext2_inode *inode);
extern int ext2fs_inode_has_valid_blocks2(ext2_filsys fs,

```



```

    struct ext2_inode *inode);

/* version.c */
extern int ext2fs_parse_version_string(const char *ver_string);
extern int ext2fs_get_library_version(const char **ver_string,
    const char **date_string);

/* write_bb_file.c */
extern errcode_t ext2fs_write_bb_FILE(ext2_badblocks_list bb_list,
    unsigned int flags,
    FILE *f);

/* inline functions */
#ifdef NO_INLINE_FUNCS
extern errcode_t ext2fs_get_mem(unsigned long size, void *ptr);
extern errcode_t ext2fs_get_memzero(unsigned long size, void *ptr);
extern errcode_t ext2fs_get_array(unsigned long count,
    unsigned long size, void *ptr);
extern errcode_t ext2fs_get_arrayzero(unsigned long count,
    unsigned long size, void *ptr);
extern errcode_t ext2fs_free_mem(void *ptr);
extern errcode_t ext2fs_resize_mem(unsigned long old_size,
    unsigned long size, void *ptr);
extern void ext2fs_mark_super_dirty(ext2_filsys fs);
extern void ext2fs_mark_changed(ext2_filsys fs);
extern int ext2fs_test_changed(ext2_filsys fs);
extern void ext2fs_mark_valid(ext2_filsys fs);
extern void ext2fs_unmark_valid(ext2_filsys fs);
extern int ext2fs_test_valid(ext2_filsys fs);
extern void ext2fs_mark_ib_dirty(ext2_filsys fs);
extern void ext2fs_mark_bb_dirty(ext2_filsys fs);
extern int ext2fs_test_ib_dirty(ext2_filsys fs);
extern int ext2fs_test_bb_dirty(ext2_filsys fs);
extern dgrp_t ext2fs_group_of_blk(ext2_filsys fs, blk_t blk);
extern dgrp_t ext2fs_group_of_ino(ext2_filsys fs, ext2_ino_t ino);
extern blk_t ext2fs_group_first_block(ext2_filsys fs, dgrp_t group);
extern blk_t ext2fs_group_last_block(ext2_filsys fs, dgrp_t group);
extern blk_t ext2fs_inode_data_blocks(ext2_filsys fs,
    struct ext2_inode *inode);
extern unsigned int ext2fs_div_ceil(unsigned int a, unsigned int b);
extern __u64 ext2fs_div64_ceil(__u64 a, __u64 b);
extern int ext2fs_dirent_name_len(const struct ext2_dir_entry *entry);
extern void ext2fs_dirent_set_name_len(struct ext2_dir_entry *entry, int len);
extern int ext2fs_dirent_file_type(const struct ext2_dir_entry *entry);
extern void ext2fs_dirent_set_file_type(struct ext2_dir_entry *entry, int type);
extern struct ext2_inode *ext2fs_inode(struct ext2_inode_large *large_inode);
extern const struct ext2_inode *ext2fs_const_inode(const struct ext2_inode_large *large_inode);

#endif

/*
 * The actual inlined functions definitions themselves...
 */

```

```

* If NO_INLINE_FUNCS is defined, then we won't try to do inline
* functions at all!
*/
#if (defined(INCLUDE_INLINE_FUNCS) || !defined(NO_INLINE_FUNCS))
#ifdef INCLUDE_INLINE_FUNCS
#define _INLINE_ extern
#else
#if (__STDC_VERSION__ >= 199901L)
#define _INLINE_ inline
#else
#ifdef __GNUC__
#define _INLINE_ extern __inline__
#else/* For Watcom C */
#define _INLINE_ extern inline
#endif /* __GNUC__ */
#endif /* __STDC_VERSION__ >= 199901L */
#endif

#ifdef EXT2_CUSTOM_MEMORY_ROUTINES
#include <string.h>
/*
* Allocate memory. The 'ptr' arg must point to a pointer.
*/
_INLINE_ errcode_t ext2fs_get_mem(unsigned long size, void *ptr)
{
void *pp;

pp = malloc(size);
if (!pp)
return EXT2_ET_NO_MEMORY;
memcpy(ptr, &pp, sizeof(pp));
return 0;
}

_INLINE_ errcode_t ext2fs_get_memzero(unsigned long size, void *ptr)
{
void *pp;

pp = malloc(size);
if (!pp)
return EXT2_ET_NO_MEMORY;
memset(pp, 0, size);
memcpy(ptr, &pp, sizeof(pp));
return 0;
}

_INLINE_ errcode_t ext2fs_get_array(unsigned long count, unsigned long size, void *ptr)
{
if (count && (~0UL)/count < size)
return EXT2_ET_NO_MEMORY;
return ext2fs_get_mem(count*size, ptr);
}

_INLINE_ errcode_t ext2fs_get_arrayzero(unsigned long count,
unsigned long size, void *ptr)

```

```

{
void *pp;

if (count && (~0UL)/count < size)
return EXT2_ET_NO_MEMORY;
pp = calloc(count, size);
if (!pp)
return EXT2_ET_NO_MEMORY;
memcpy(ptr, &pp, sizeof(pp));
return 0;
}

/*
 * Free memory. The 'ptr' arg must point to a pointer.
 */
_INLINE_ errcode_t ext2fs_free_mem(void *ptr)
{
void *p;

memcpy(&p, ptr, sizeof(p));
free(p);
p = 0;
memcpy(ptr, &p, sizeof(p));
return 0;
}

/*
 * Resize memory. The 'ptr' arg must point to a pointer.
 */
_INLINE_ errcode_t ext2fs_resize_mem(unsigned long EXT2FS_ATTR((unused)) old_size,
unsigned long size, void *ptr)
{
void *p;

/* Use "memcpy" for pointer assignments here to avoid problems
 * with C99 strict type aliasing rules. */
memcpy(&p, ptr, sizeof(p));
p = realloc(p, size);
if (!p)
return EXT2_ET_NO_MEMORY;
memcpy(ptr, &p, sizeof(p));
return 0;
}
#endif/* Custom memory routines */

/*
 * Mark a filesystem superblock as dirty
 */
_INLINE_ void ext2fs_mark_super_dirty(ext2_filsys fs)
{
fs->flags |= EXT2_FLAG_DIRTY | EXT2_FLAG_CHANGED;
}

/*

```

```
* Mark a filesystem as changed
*/
_INLINE_ void ext2fs_mark_changed(ext2_filsys fs)
{
fs->flags |= EXT2_FLAG_CHANGED;
}

/*
 * Check to see if a filesystem has changed
 */
_INLINE_ int ext2fs_test_changed(ext2_filsys fs)
{
return (fs->flags & EXT2_FLAG_CHANGED);
}

/*
 * Mark a filesystem as valid
 */
_INLINE_ void ext2fs_mark_valid(ext2_filsys fs)
{
fs->flags |= EXT2_FLAG_VALID;
}

/*
 * Mark a filesystem as NOT valid
 */
_INLINE_ void ext2fs_unmark_valid(ext2_filsys fs)
{
fs->flags &= ~EXT2_FLAG_VALID;
}

/*
 * Check to see if a filesystem is valid
 */
_INLINE_ int ext2fs_test_valid(ext2_filsys fs)
{
return (fs->flags & EXT2_FLAG_VALID);
}

/*
 * Mark the inode bitmap as dirty
 */
_INLINE_ void ext2fs_mark_ib_dirty(ext2_filsys fs)
{
fs->flags |= EXT2_FLAG_IB_DIRTY | EXT2_FLAG_CHANGED;
}

/*
 * Mark the block bitmap as dirty
 */
_INLINE_ void ext2fs_mark_bb_dirty(ext2_filsys fs)
{
fs->flags |= EXT2_FLAG_BB_DIRTY | EXT2_FLAG_CHANGED;
}
```

```
/*
 * Check to see if a filesystem's inode bitmap is dirty
 */
_INLINE_ int ext2fs_test_ib_dirty(ext2_filsys fs)
{
return (fs->flags & EXT2_FLAG_IB_DIRTY);
}

/*
 * Check to see if a filesystem's block bitmap is dirty
 */
_INLINE_ int ext2fs_test_bb_dirty(ext2_filsys fs)
{
return (fs->flags & EXT2_FLAG_BB_DIRTY);
}

/*
 * Return the group # of a block
 */
_INLINE_ dgrp_t ext2fs_group_of_blk(ext2_filsys fs, blk_t blk)
{
return ext2fs_group_of_blk2(fs, blk);
}

/*
 * Return the group # of an inode number
 */
_INLINE_ dgrp_t ext2fs_group_of_ino(ext2_filsys fs, ext2_ino_t ino)
{
return (ino - 1) / fs->super->s_inodes_per_group;
}

/*
 * Return the first block (inclusive) in a group
 */
_INLINE_ blk_t ext2fs_group_first_block(ext2_filsys fs, dgrp_t group)
{
return (blk_t) ext2fs_group_first_block2(fs, group);
}

/*
 * Return the last block (inclusive) in a group
 */
_INLINE_ blk_t ext2fs_group_last_block(ext2_filsys fs, dgrp_t group)
{
return (blk_t) ext2fs_group_last_block2(fs, group);
}

_INLINE_ blk_t ext2fs_inode_data_blocks(ext2_filsys fs,
struct ext2_inode *inode)
{
return (blk_t) ext2fs_inode_data_blocks2(fs, inode);
}

/*
```

```

    * This is an efficient, overflow safe way of calculating ceil((1.0 * a) / b)
    */
    _INLINE_ unsigned int ext2fs_div_ceil(unsigned int a, unsigned int b)
    {
        if (!a)
            return 0;
        return ((a - 1) / b) + 1;
    }

    _INLINE_ __u64 ext2fs_div64_ceil(__u64 a, __u64 b)
    {
        if (!a)
            return 0;
        return ((a - 1) / b) + 1;
    }

    _INLINE_ int ext2fs_dirent_name_len(const struct ext2_dir_entry *entry)
    {
        return entry->name_len & 0xff;
    }

    _INLINE_ void ext2fs_dirent_set_name_len(struct ext2_dir_entry *entry, int len)
    {
        entry->name_len = (entry->name_len & 0xff00) | (len & 0xff);
    }

    _INLINE_ int ext2fs_dirent_file_type(const struct ext2_dir_entry *entry)
    {
        return entry->name_len >> 8;
    }

    _INLINE_ void ext2fs_dirent_set_file_type(struct ext2_dir_entry *entry, int type)
    {
        entry->name_len = (entry->name_len & 0xff) | (type << 8);
    }

    _INLINE_ struct ext2_inode *ext2fs_inode(struct ext2_inode_large *large_inode)
    {
        /* It is always safe to convert large inode to a small inode */
        return (struct ext2_inode *) large_inode;
    }

    _INLINE_ const struct ext2_inode *
    ext2fs_const_inode(const struct ext2_inode_large *large_inode)
    {
        /* It is always safe to convert large inode to a small inode */
        return (const struct ext2_inode *) large_inode;
    }

    #undef _INLINE_
    #endif

    #ifdef __cplusplus
    }
    #endif

```

```
#endif /* _EXT2FS_EXT2FS_H */
```

This package, the EXT2 filesystem utilities, are made available under the GNU Public License version 2, with the exception of the lib/ext2fs and lib/e2p libraries, which are made available under the GNU Library General Public License Version 2, the lib/uuid library which is made available under a BSD-style license and the lib/et and lib/ss libraries which are made available under an MIT-style license. Please see lib/uuid/COPYING for more details for the license for the files comprising the libuuid library, and the source file headers of the libet and libss libraries for more information.

The most recent officially distributed version can be found at <http://e2fsprogs.sourceforge.net>. If you need to make a distribution, that's the one you should use. If there is some reason why you'd like a more recent version that is still in ALPHA testing (i.e., either using the "WIP" test distributions or one from the hg or git repository from the development branch, please contact me (tytso@mit.edu) before you ship. The release schedules for this package are flexible, if you give me enough lead time.

Theodore Ts'o
23-June-2007

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the

notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to

these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free

Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is

the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is

included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a

work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are

prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time.

Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is

safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

```
/*
 * Copyright 1987, 1988 by MIT Student Information Processing Board
 *
 * Permission to use, copy, modify, and distribute this software and
 * its documentation for any purpose is hereby granted, provided that
 * the names of M.I.T. and the M.I.T. S.I.P.B. not be used in
 * advertising or publicity pertaining to distribution of the software
 * without specific, written prior permission. M.I.T. and the
 * M.I.T. S.I.P.B. make no representations about the suitability of
 * this software for any purpose. It is provided "as is" without
 * express or implied warranty.
 *
 * This quote is just too good to not pass on:
 *
 * "BTW, I would have rejected the name Story Server because its
 * initials are SS, the name of the secret police in Nazi
 * Germany, probably the most despised pair of letters in western
 * culture." --- http://scriptingnewsarchive.userland.com/1999/12/13
 *
 * Let no one say political correctness isn't dead....
 */
```

```

#ifndef __ss_h
#define __ss_h __FILE__

#include <ss/ss_err.h>

#define __SS_CONST const
#define __SS_PROTO (int, const char * const *, int, void *)

#ifdef __GNUC__
#define __SS_ATTR(x) __attribute__((x))
#else
#define __SS_ATTR(x)
#endif

typedef __SS_CONST struct ss_request_entry {
    __SS_CONST char * __SS_CONST *command_names; /* whatever */
    void (* __SS_CONST function) __SS_PROTO; /* foo */
    __SS_CONST char * __SS_CONST info_string; /* NULL */
    int flags; /* 0 */
} ss_request_entry;

typedef __SS_CONST struct ss_request_table {
    int version;
    ss_request_entry *requests;
} ss_request_table;

#define SS_RQT_TBL_V22

typedef struct ss_rp_options { /* DEFAULT VALUES */
    int version; /* SS_RP_V1 */
    void (*unknown) __SS_PROTO; /* call for unknown command */
    int allow_suspend;
    int catch_int;
} ss_rp_options;

#define SS_RP_V1 1

#define SS_OPT_DONT_LIST 0x0001
#define SS_OPT_DONT_SUMMARIZE 0x0002

void ss_help __SS_PROTO;
#if 0
char *ss_current_request(); /* This is actually a macro */
#endif

char *ss_name(int sci_idx);
void ss_error (int, long, char const *, ...)
__SS_ATTR((format(printf, 3, 4)));
void ss_perror (int, long, char const *);

int ss_create_invocation(const char *, const char *, void *,
    ss_request_table *, int *);

```

```

void ss_delete_invocation(int);
int ss_listen(int);
int ss_execute_line(int, char *);
void ss_add_request_table(int, ss_request_table *, int, int *);
void ss_delete_request_table(int, ss_request_table *, int *);
void ss_abort_subsystem(int sci_idx, int code);
void ss_quit(int argc, const char * const *argv, int sci_idx, void *infop);
void ss_self_identify(int argc, const char * const *argv, int sci_idx, void *infop);
void ss_subsystem_name(int argc, const char * const *argv,
    int sci_idx, void *infop);
void ss_subsystem_version(int argc, const char * const *argv,
    int sci_idx, void *infop);
void ss_unimplemented(int argc, const char * const *argv,
    int sci_idx, void *infop);
void ss_set_prompt(int sci_idx, char *new_prompt);
char *ss_get_prompt(int sci_idx);
void ss_get_readline(int sci_idx);
char *ss_safe_getenv(const char *arg);

extern ss_request_table ss_std_requests;
#endif /* _ss_h */
/*
 * Public include file for the UUID library
 *
 * Copyright (C) 1996, 1997, 1998 Theodore Ts'o.
 *
 * %Begin-Header%
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, and the entire permission notice in its entirety,
 *    including the disclaimer of warranties.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. The name of the author may not be used to endorse or promote
 *    products derived from this software without specific prior
 *    written permission.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ALL OF
 * WHICH ARE HEREBY DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
 * OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
 * BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
 * USE OF THIS SOFTWARE, EVEN IF NOT ADVISED OF THE POSSIBILITY OF SUCH
 * DAMAGE.
 * %End-Header%
 */

```



```
#ifndef _UUID_UUID_H
#define _UUID_UUID_H

#include <sys/types.h>
#ifndef _WIN32
#include <sys/time.h>
#endif
#include <time.h>

typedef unsigned char uuid_t[16];

/* UUID Variant definitions */
#define UUID_VARIANT_NCS 0
#define UUID_VARIANT_DCE 1
#define UUID_VARIANT_MICROSOFT2
#define UUID_VARIANT_OTHER3

/* UUID Type definitions */
#define UUID_TYPE_DCE_TIME 1
#define UUID_TYPE_DCE_RANDOM 4

/* Allow UUID constants to be defined */
#ifdef __GNUC__
#define UUID_DEFINE(name,u0,u1,u2,u3,u4,u5,u6,u7,u8,u9,u10,u11,u12,u13,u14,u15) \
static const uuid_t name __attribute__((unused)) = {u0,u1,u2,u3,u4,u5,u6,u7,u8,u9,u10,u11,u12,u13,u14,u15}
#else
#define UUID_DEFINE(name,u0,u1,u2,u3,u4,u5,u6,u7,u8,u9,u10,u11,u12,u13,u14,u15) \
static const uuid_t name = {u0,u1,u2,u3,u4,u5,u6,u7,u8,u9,u10,u11,u12,u13,u14,u15}
#endif

#ifdef __cplusplus
extern "C" {
#endif

/* clear.c */
void uuid_clear(uuid_t uu);

/* compare.c */
int uuid_compare(const uuid_t uu1, const uuid_t uu2);

/* copy.c */
void uuid_copy(uuid_t dst, const uuid_t src);

/* gen_uuid.c */
void uuid_generate(uuid_t out);
void uuid_generate_random(uuid_t out);
void uuid_generate_time(uuid_t out);

/* isnull.c */
int uuid_is_null(const uuid_t uu);

/* parse.c */
int uuid_parse(const char *in, uuid_t uu);
```

```
/* unparse.c */
void uuid_unparse(const uuid_t uu, char *out);
void uuid_unparse_lower(const uuid_t uu, char *out);
void uuid_unparse_upper(const uuid_t uu, char *out);

/* uuid_time.c */
time_t uuid_time(const uuid_t uu, struct timeval *ret_tv);
int uuid_type(const uuid_t uu);
int uuid_variant(const uuid_t uu);

#ifdef __cplusplus
}
#endif

#endif /* _UUID_UUID_H */
```

1.28 ebttables

1.28.1 Available under license

All code in this package, including the code from the extensions, is released under the GPL license, which you find hereafter.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free

software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any

part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES

PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

```
This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
```


under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.29 elfutils

1.29.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.30 ethtool

1.30.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
/*
 * ethtool.c: Linux ethernet device configuration tool.
 *
 * Copyright (C) 1998 David S. Miller (davem@dm.cobaltmicro.com)
 * Portions Copyright 2001 Sun Microsystems
 * Kernel 2.4 update Copyright 2001 Jeff Garzik <jgarzik@mandrakesoft.com>
 * Wake-on-LAN,natsemi,misc support by Tim Hockin <thockin@sun.com>
 * Portions Copyright 2002 Intel
 * Portions Copyright (C) Sun Microsystems 2008
 * do_test support by Eli Kupermann <eli.kupermann@intel.com>
 * ETHTOOL_PHYS_ID support by Chris Leech <christopher.leech@intel.com>
 * e1000 support by Scott Feldman <scott.feldman@intel.com>
 * e100 support by Wen Tao <wen-hwa.tao@intel.com>
 * ixgb support by Nicholas Nunley <Nicholas.d.nunley@intel.com>
 * amd8111e support by Reeja John <reeja.john@amd.com>
 * long arguments by Andi Kleen.
 * SMSC LAN911x support by Steve Glendinning <steve.glendinning@smc.com>
 * Rx Network Flow Control configuration support <santwona.behera@sun.com>
 * Various features by Ben Hutchings <bhutchings@solarflare.com>;
 * Copyright 2009, 2010 Solarflare Communications
 * MDI-X set support by Jesse Brandeburg <jesse.brandeburg@intel.com>
 * Copyright 2012 Intel Corporation
 * vmxnet3 support by Shrikrishna Khare <skhare@vmware.com>
 * Various features by Ben Hutchings <ben@decadent.org.uk>;
 * Copyright 2008-2010, 2013-2016 Ben Hutchings
 *
 * TODO:
 * * show settings for all devices
 */

#include "internal.h"
#include <string.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <stdio.h>
```

```

#include <stddef.h>
#include <stdbool.h>
#include <errno.h>
#include <sys/utsname.h>
#include <limits.h>
#include <ctype.h>

#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#include <linux/sockios.h>
#include <linux/netlink.h>

#ifndef MAX_ADDR_LEN
#define MAX_ADDR_LEN32
#endif

#ifndef HAVE_NETIF_MSG
enum {
    NETIF_MSG_DRV= 0x0001,
    NETIF_MSG_PROBE= 0x0002,
    NETIF_MSG_LINK= 0x0004,
    NETIF_MSG_TIMER= 0x0008,
    NETIF_MSG_IFDOWN= 0x0010,
    NETIF_MSG_IFUP= 0x0020,
    NETIF_MSG_RX_ERR= 0x0040,
    NETIF_MSG_TX_ERR= 0x0080,
    NETIF_MSG_TX_QUEUED= 0x0100,
    NETIF_MSG_INTR= 0x0200,
    NETIF_MSG_TX_DONE= 0x0400,
    NETIF_MSG_RX_STATUS= 0x0800,
    NETIF_MSG_PKTDATA= 0x1000,
    NETIF_MSG_HW= 0x2000,
    NETIF_MSG_WOL= 0x4000,
};
#endif

#ifndef NETLINK_GENERIC
#define NETLINK_GENERIC16
#endif

#define KERNEL_VERSION(a,b,c) (((a) << 16) + ((b) << 8) + (c))

static void exit_bad_args(void) __attribute__((noreturn));

static void exit_bad_args(void)
{
    fprintf(stderr,
    "ethtool: bad command line argument(s)\n"
    "For more information run ethtool -h\n");
    exit(1);
}

typedef enum {

```

```

CMDL_NONE,
CMDL_BOOL,
CMDL_S32,
CMDL_U8,
CMDL_U16,
CMDL_U32,
CMDL_U64,
CMDL_BE16,
CMDL_IP4,
CMDL_STR,
CMDL_FLAG,
CMDL_MAC,
} cmdline_type_t;

struct cmdline_info {
const char *name;
cmdline_type_t type;
/* Points to int (BOOL), s32, u16, u32 (U32/FLAG/IP4), u64,
 * char * (STR) or u8[6] (MAC). For FLAG, the value accumulates
 * all flags to be set. */
void *wanted_val;
void *ioctl_val;
/* For FLAG, the flag value to be set/cleared */
u32 flag_val;
/* For FLAG, points to u32 and accumulates all flags seen.
 * For anything else, points to int and is set if the option is
 * seen. */
void *seen_val;
};

struct flag_info {
const char *name;
u32 value;
};

static const struct flag_info flags_msglvl[] = {
{ "drv", NETIF_MSG_DRV },
{ "probe", NETIF_MSG_PROBE },
{ "link", NETIF_MSG_LINK },
{ "timer", NETIF_MSG_TIMER },
{ "ifdown", NETIF_MSG_IFDOWN },
{ "ifup", NETIF_MSG_IFUP },
{ "rx_err", NETIF_MSG_RX_ERR },
{ "tx_err", NETIF_MSG_TX_ERR },
{ "tx_queued", NETIF_MSG_TX_QUEUED },
{ "intr", NETIF_MSG_INTR },
{ "tx_done", NETIF_MSG_TX_DONE },
{ "rx_status", NETIF_MSG_RX_STATUS },
{ "pktdata", NETIF_MSG_PKTDATA },
{ "hw", NETIF_MSG_HW },
{ "wol", NETIF_MSG_WOL },
};

struct off_flag_def {

```

```

const char *short_name;
const char *long_name;
const char *kernel_name;
u32 get_cmd, set_cmd;
u32 value;
/* For features exposed through ETHTOOL_GFLAGS, the oldest
 * kernel version for which we can trust the result. Where
 * the flag was added at the same time the kernel started
 * supporting the feature, this is 0 (to allow for backports).
 * Where the feature was supported before the flag was added,
 * it is the version that introduced the flag.
 */
u32 min_kernel_ver;
};
static const struct off_flag_def off_flag_def[] = {
{ "rx", "rx-checksumming", "rx-checksum",
  ETHTOOL_GRXCSUM, ETHTOOL_SRXCSUM, ETH_FLAG_RXCSUM, 0 },
{ "tx", "tx-checksumming", "tx-checksum-*",
  ETHTOOL_GTXCSUM, ETHTOOL_STXCSUM, ETH_FLAG_TXCSUM, 0 },
{ "sg", "scatter-gather", "tx-scatter-gather*",
  ETHTOOL_GSG, ETHTOOL_SSG, ETH_FLAG_SG, 0 },
{ "tso", "tcp-segmentation-offload", "tx-tcp*-segmentation",
  ETHTOOL_GTSO, ETHTOOL_STSO, ETH_FLAG_TSO, 0 },
{ "ufo", "udp-fragmentation-offload", "tx-udp-fragmentation",
  ETHTOOL_GUFO, ETHTOOL_SUFO, ETH_FLAG_UFO, 0 },
{ "gso", "generic-segmentation-offload", "tx-generic-segmentation",
  ETHTOOL_GGSO, ETHTOOL_SGSO, ETH_FLAG_GSO, 0 },
{ "gro", "generic-receive-offload", "rx-gro",
  ETHTOOL_GGRO, ETHTOOL_SGRO, ETH_FLAG_GRO, 0 },
{ "lro", "large-receive-offload", "rx-lro",
  0, 0, ETH_FLAG_LRO,
  KERNEL_VERSION(2,6,24) },
{ "rxvlan", "rx-vlan-offload", "rx-vlan-hw-parse",
  0, 0, ETH_FLAG_RXVLAN,
  KERNEL_VERSION(2,6,37) },
{ "txvlan", "tx-vlan-offload", "tx-vlan-hw-insert",
  0, 0, ETH_FLAG_TXVLAN,
  KERNEL_VERSION(2,6,37) },
{ "ntuple", "ntuple-filters", "rx-ntuple-filter",
  0, 0, ETH_FLAG_NTUPLE, 0 },
{ "rxhash", "receive-hashing", "rx-hashing",
  0, 0, ETH_FLAG_RXHASH, 0 },
};

struct feature_def {
char name[ETH_GSTRING_LEN];
int off_flag_index; /* index in off_flag_def; negative if none match */
};

struct feature_defs {
size_t n_features;
/* Number of features each offload flag is associated with */
unsigned int off_flag_matched[ARRAY_SIZE(off_flag_def)];
/* Name and offload flag index for each feature */
struct feature_def def[0];
};

```



```

};

#define FEATURE_BITS_TO_BLOCKS(n_bits)DIV_ROUND_UP(n_bits, 32U)
#define FEATURE_WORD(blocks, index, field)((blocks)[(index) / 32U].field)
#define FEATURE_FIELD_FLAG(index)(1U << (index) % 32U)
#define FEATURE_BIT_SET(blocks, index, field)\
(FEATURE_WORD(blocks, index, field) |= FEATURE_FIELD_FLAG(index))
#define FEATURE_BIT_CLEAR(blocks, index, field)\
(FEATURE_WORD(blocks, index, field) &= ~FEATURE_FIELD_FLAG(index))
#define FEATURE_BIT_IS_SET(blocks, index, field)\
(FEATURE_WORD(blocks, index, field) & FEATURE_FIELD_FLAG(index))

static long long
get_int_range(char *str, int base, long long min, long long max)
{
    long long v;
    char *endp;

    if (!str)
        exit_bad_args();
    errno = 0;
    v = strtoll(str, &endp, base);
    if (errno || *endp || v < min || v > max)
        exit_bad_args();
    return v;
}

static unsigned long long
get_uint_range(char *str, int base, unsigned long long max)
{
    unsigned long long v;
    char *endp;

    if (!str)
        exit_bad_args();
    errno = 0;
    v = strtoull(str, &endp, base);
    if (errno || *endp || v > max)
        exit_bad_args();
    return v;
}

static int get_int(char *str, int base)
{
    return get_int_range(str, base, INT_MIN, INT_MAX);
}

static u32 get_u32(char *str, int base)
{
    return get_uint_range(str, base, 0xffffffff);
}

static void get_mac_addr(char *src, unsigned char *dest)
{

```

```

int count;
int i;
int buf[ETH_ALEN];

count = sscanf(src, "%2x:%2x:%2x:%2x:%2x:%2x",
&buf[0], &buf[1], &buf[2], &buf[3], &buf[4], &buf[5]);
if (count != ETH_ALEN)
exit_bad_args();

for (i = 0; i < count; i++) {
dest[i] = buf[i];
}
}

static int parse_hex_u32_bitmap(const char *s,
unsigned int nbits, u32 *result)
{
const unsigned int nwords = __KERNEL_DIV_ROUND_UP(nbits, 32);
size_t slen = strlen(s);
size_t i;

/* ignore optional '0x' prefix */
if ((slen > 2) && (strncasecmp(s, "0x", 2) == 0)) {
slen -= 2;
s += 2;
}

if (slen > 8 * nwords) /* up to 2 digits per byte */
return -1;

memset(result, 0, 4 * nwords);
for (i = 0; i < slen; ++i) {
const unsigned int shift = (slen - 1 - i) * 4;
u32 *dest = &result[shift / 32];
u32 nibble;

if ('a' <= s[i] && s[i] <= 'f')
nibble = 0xa + (s[i] - 'a');
else if ('A' <= s[i] && s[i] <= 'F')
nibble = 0xa + (s[i] - 'A');
else if ('0' <= s[i] && s[i] <= '9')
nibble = (s[i] - '0');
else
return -1;

*dest |= (nibble << (shift % 32));
}

return 0;
}

static void parse_generic_cmdline(struct cmd_context *ctx,
int *changed,
struct cmdline_info *info,
unsigned int n_info)

```

```
{
int argc = ctx->argc;
char **argv = ctx->argv;
int i, idx;
int found;

for (i = 0; i < argc; i++) {
found = 0;
for (idx = 0; idx < n_info; idx++) {
if (!strcmp(info[idx].name, argv[i])) {
found = 1;
*changed = 1;
if (info[idx].type != CMDL_FLAG &&
    info[idx].seen_val)
*(int *)info[idx].seen_val = 1;
i += 1;
if (i >= argc)
exit_bad_args();
switch (info[idx].type) {
case CMDL_BOOL: {
int *p = info[idx].wanted_val;
if (!strcmp(argv[i], "on"))
*p = 1;
else if (!strcmp(argv[i], "off"))
*p = 0;
else
exit_bad_args();
break;
}
case CMDL_S32: {
s32 *p = info[idx].wanted_val;
*p = get_int_range(argv[i], 0,
    -0x80000000LL,
    0x7fffffff);
break;
}
case CMDL_U8: {
u8 *p = info[idx].wanted_val;
*p = get_uint_range(argv[i], 0, 0xff);
break;
}
case CMDL_U16: {
u16 *p = info[idx].wanted_val;
*p = get_uint_range(argv[i], 0, 0xffff);
break;
}
case CMDL_U32: {
u32 *p = info[idx].wanted_val;
*p = get_uint_range(argv[i], 0,
    0xffffffff);
break;
}
case CMDL_U64: {
u64 *p = info[idx].wanted_val;
```

```

    *p = get_uint_range(
    argp[i], 0,
    0xffffffffffffLL);
    break;
}
case CMDL_BE16: {
    u16 *p = info[idx].wanted_val;
    *p = cpu_to_be16(
    get_uint_range(argp[i], 0,
    0xffff));
    break;
}
case CMDL_IP4: {
    u32 *p = info[idx].wanted_val;
    struct in_addr in;
    if (!inet_aton(argp[i], &in))
        exit_bad_args();
    *p = in.s_addr;
    break;
}
case CMDL_MAC:
    get_mac_addr(argp[i],
    info[idx].wanted_val);
    break;
case CMDL_FLAG: {
    u32 *p;
    p = info[idx].seen_val;
    *p |= info[idx].flag_val;
    if (!strcmp(argp[i], "on")) {
        p = info[idx].wanted_val;
        *p |= info[idx].flag_val;
    } else if (strcmp(argp[i], "off")) {
        exit_bad_args();
    }
    break;
}
case CMDL_STR: {
    char **s = info[idx].wanted_val;
    *s = strdup(argp[i]);
    break;
}
default:
    exit_bad_args();
}
break;
}
}
if( !found)
    exit_bad_args();
}
}

static void flag_to_cmdline_info(const char *name, u32 value,
    u32 *wanted, u32 *mask,
    struct cmdline_info *cli)

```

```

{
memset(cli, 0, sizeof(*cli));
cli->name = name;
cli->type = CMDL_FLAG;
cli->flag_val = value;
cli->wanted_val = wanted;
cli->seen_val = mask;
}

static void
print_flags(const struct flag_info *info, unsigned int n_info, u32 value)
{
const char *sep = "";

while (n_info) {
if (value & info->value) {
printf("%s%s", sep, info->name);
sep = " ";
value &= ~info->value;
}
++info;
--n_info;
}

/* Print any unrecognised flags in hex */
if (value)
printf("%s%#x", sep, value);
}

static int rxflow_str_to_type(const char *str)
{
int flow_type = 0;

if (!strcmp(str, "tcp4"))
flow_type = TCP_V4_FLOW;
else if (!strcmp(str, "udp4"))
flow_type = UDP_V4_FLOW;
else if (!strcmp(str, "ah4") || !strcmp(str, "esp4"))
flow_type = AH_ESP_V4_FLOW;
else if (!strcmp(str, "sctp4"))
flow_type = SCTP_V4_FLOW;
else if (!strcmp(str, "tcp6"))
flow_type = TCP_V6_FLOW;
else if (!strcmp(str, "udp6"))
flow_type = UDP_V6_FLOW;
else if (!strcmp(str, "ah6") || !strcmp(str, "esp6"))
flow_type = AH_ESP_V6_FLOW;
else if (!strcmp(str, "sctp6"))
flow_type = SCTP_V6_FLOW;
else if (!strcmp(str, "ether"))
flow_type = ETHER_FLOW;

return flow_type;
}

```

```

static int do_version(struct cmd_context *ctx)
{
    fprintf(stdout,
    PACKAGE " version " VERSION
#ifdef ETHTOOL_ENABLE_PRETTY_DUMP
    " (pretty dumps disabled)"
#endif
    "\n");
    return 0;
}

/* link mode routines */

static ETHTOOL_DECLARE_LINK_MODE_MASK(all_advertised_modes);
static ETHTOOL_DECLARE_LINK_MODE_MASK(all_advertised_flags);

static void init_global_link_mode_masks(void)
{
    static const enum ethtool_link_mode_bit_indices
    all_advertised_modes_bits[] = {
        ETHTOOL_LINK_MODE_10baseT_Half_BIT,
        ETHTOOL_LINK_MODE_10baseT_Full_BIT,
        ETHTOOL_LINK_MODE_100baseT_Half_BIT,
        ETHTOOL_LINK_MODE_100baseT_Full_BIT,
        ETHTOOL_LINK_MODE_1000baseT_Half_BIT,
        ETHTOOL_LINK_MODE_1000baseT_Full_BIT,
        ETHTOOL_LINK_MODE_1000baseKX_Full_BIT,
        ETHTOOL_LINK_MODE_2500baseX_Full_BIT,
        ETHTOOL_LINK_MODE_10000baseT_Full_BIT,
        ETHTOOL_LINK_MODE_10000baseKX4_Full_BIT,
        ETHTOOL_LINK_MODE_10000baseKR_Full_BIT,
        ETHTOOL_LINK_MODE_10000baseR_FEC_BIT,
        ETHTOOL_LINK_MODE_20000baseMLD2_Full_BIT,
        ETHTOOL_LINK_MODE_20000baseKR2_Full_BIT,
        ETHTOOL_LINK_MODE_40000baseKR4_Full_BIT,
        ETHTOOL_LINK_MODE_40000baseCR4_Full_BIT,
        ETHTOOL_LINK_MODE_40000baseSR4_Full_BIT,
        ETHTOOL_LINK_MODE_40000baseLR4_Full_BIT,
        ETHTOOL_LINK_MODE_56000baseKR4_Full_BIT,
        ETHTOOL_LINK_MODE_56000baseCR4_Full_BIT,
        ETHTOOL_LINK_MODE_56000baseSR4_Full_BIT,
        ETHTOOL_LINK_MODE_56000baseLR4_Full_BIT,
    };
    static const enum ethtool_link_mode_bit_indices
    additional_advertised_flags_bits[] = {
        ETHTOOL_LINK_MODE_Autoneg_BIT,
        ETHTOOL_LINK_MODE_TP_BIT,
        ETHTOOL_LINK_MODE_AUI_BIT,
        ETHTOOL_LINK_MODE_MII_BIT,
        ETHTOOL_LINK_MODE_FIBRE_BIT,
        ETHTOOL_LINK_MODE_BNC_BIT,
        ETHTOOL_LINK_MODE_Pause_BIT,
        ETHTOOL_LINK_MODE_Asym_Pause_BIT,
        ETHTOOL_LINK_MODE_Backplane_BIT,
    };
}

```

```

};
unsigned int i;

ethtool_link_mode_zero(all_advertised_modes);
ethtool_link_mode_zero(all_advertised_flags);
for (i = 0; i < ARRAY_SIZE(all_advertised_modes_bits); ++i) {
ethtool_link_mode_set_bit(all_advertised_modes_bits[i],
    all_advertised_modes);
ethtool_link_mode_set_bit(all_advertised_modes_bits[i],
    all_advertised_flags);
}

for (i = 0; i < ARRAY_SIZE(additional_advertised_flags_bits); ++i) {
ethtool_link_mode_set_bit(
    additional_advertised_flags_bits[i],
    all_advertised_flags);
}
}

static void dump_link_caps(const char *prefix, const char *an_prefix,
    const u32 *mask, int link_mode_only);

static void dump_supported(const struct ethtool_link_usettings *link_usettings)
{
fprintf(stdout, "Supported ports: [ ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_TP_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "TP ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_AUI_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "AUI ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_BNC_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "BNC ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_MII_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "MII ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_FIBRE_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "FIBRE ");
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_Backplane_BIT,
    link_usettings->link_modes.supported))
fprintf(stdout, "Backplane ");
fprintf(stdout, "]\n");

dump_link_caps("Supported", "Supports",
    link_usettings->link_modes.supported, 0);
}

```

```

/* Print link capability flags (supported, advertised or lp_advertised).
 * Assumes that the corresponding SUPPORTED and ADVERTISED flags are equal.
 */
static void dump_link_caps(const char *prefix, const char *an_prefix,
    const u32 *mask, int link_mode_only)
{
    static const struct {
        int same_line; /* print on same line as previous */
        unsigned int bit_index;
        const char *name;
    } mode_defs[] = {
        { 0, ETHTOOL_LINK_MODE_10baseT_Half_BIT,
          "10baseT/Half" },
        { 1, ETHTOOL_LINK_MODE_10baseT_Full_BIT,
          "10baseT/Full" },
        { 0, ETHTOOL_LINK_MODE_100baseT_Half_BIT,
          "100baseT/Half" },
        { 1, ETHTOOL_LINK_MODE_100baseT_Full_BIT,
          "100baseT/Full" },
        { 0, ETHTOOL_LINK_MODE_1000baseT_Half_BIT,
          "1000baseT/Half" },
        { 1, ETHTOOL_LINK_MODE_1000baseT_Full_BIT,
          "1000baseT/Full" },
        { 0, ETHTOOL_LINK_MODE_1000baseKX_Full_BIT,
          "1000baseKX/Full" },
        { 0, ETHTOOL_LINK_MODE_2500baseX_Full_BIT,
          "2500baseX/Full" },
        { 0, ETHTOOL_LINK_MODE_10000baseT_Full_BIT,
          "10000baseT/Full" },
        { 0, ETHTOOL_LINK_MODE_10000baseKX4_Full_BIT,
          "10000baseKX4/Full" },
        { 0, ETHTOOL_LINK_MODE_10000baseKR_Full_BIT,
          "10000baseKR/Full" },
        { 0, ETHTOOL_LINK_MODE_20000baseMLD2_Full_BIT,
          "20000baseMLD2/Full" },
        { 0, ETHTOOL_LINK_MODE_20000baseKR2_Full_BIT,
          "20000baseKR2/Full" },
        { 0, ETHTOOL_LINK_MODE_40000baseKR4_Full_BIT,
          "40000baseKR4/Full" },
        { 0, ETHTOOL_LINK_MODE_40000baseCR4_Full_BIT,
          "40000baseCR4/Full" },
        { 0, ETHTOOL_LINK_MODE_40000baseSR4_Full_BIT,
          "40000baseSR4/Full" },
        { 0, ETHTOOL_LINK_MODE_40000baseLR4_Full_BIT,
          "40000baseLR4/Full" },
        { 0, ETHTOOL_LINK_MODE_56000baseKR4_Full_BIT,
          "56000baseKR4/Full" },
        { 0, ETHTOOL_LINK_MODE_56000baseCR4_Full_BIT,
          "56000baseCR4/Full" },
        { 0, ETHTOOL_LINK_MODE_56000baseSR4_Full_BIT,
          "56000baseSR4/Full" },
        { 0, ETHTOOL_LINK_MODE_56000baseLR4_Full_BIT,
          "56000baseLR4/Full" },
    };
};

```



```

int indent;
int did1, new_line_pend, i;

/* Indent just like the separate functions used to */
indent = strlen(prefix) + 14;
if (indent < 24)
    indent = 24;

fprintf(stdout, "%s link modes:%*s", prefix,
        indent - (int)strlen(prefix) - 12, "");
did1 = 0;
new_line_pend = 0;
for (i = 0; i < ARRAY_SIZE(mode_defs); i++) {
    if (did1 && !mode_defs[i].same_line)
        new_line_pend = 1;
    if (ethtool_link_mode_test_bit(mode_defs[i].bit_index,
        mask)) {
        if (new_line_pend) {
            fprintf(stdout, "\n");
            fprintf(stdout, "%*s", indent, "");
            new_line_pend = 0;
        }
        did1++;
        fprintf(stdout, "%s ", mode_defs[i].name);
    }
}
if (did1 == 0)
    fprintf(stdout, "Not reported");
fprintf(stdout, "\n");

if (!link_mode_only) {
    fprintf(stdout, "%s pause frame use: ", prefix);
    if (ethtool_link_mode_test_bit(
        ETHTOOL_LINK_MODE_Pause_BIT, mask)) {
        fprintf(stdout, "Symmetric");
        if (ethtool_link_mode_test_bit(
            ETHTOOL_LINK_MODE_Asym_Pause_BIT, mask))
            fprintf(stdout, " Receive-only");
        fprintf(stdout, "\n");
    } else {
        if (ethtool_link_mode_test_bit(
            ETHTOOL_LINK_MODE_Asym_Pause_BIT, mask))
            fprintf(stdout, "Transmit-only\n");
        else
            fprintf(stdout, "No\n");
    }
}

fprintf(stdout, "%s auto-negotiation: ", an_prefix);
if (ethtool_link_mode_test_bit(
    ETHTOOL_LINK_MODE_Autoneg_BIT, mask))
    fprintf(stdout, "Yes\n");
else
    fprintf(stdout, "No\n");
}

```

```

}

static int
dump_link_usettings(const struct ethtool_link_usettings *link_usettings)
{
    dump_supported(link_usettings);
    dump_link_caps("Advertised", "Advertised",
        link_usettings->link_modes.advertising, 0);
    if (!ethtool_link_mode_is_empty(
        link_usettings->link_modes.lp_advertising))
    dump_link_caps("Link partner advertised",
        "Link partner advertised",
        link_usettings->link_modes.lp_advertising, 0);

    fprintf(stdout, "Speed: ");
    if (link_usettings->base.speed == 0
        || link_usettings->base.speed == (u16)(-1)
        || link_usettings->base.speed == (u32)(-1))
    fprintf(stdout, "Unknown!\n");
    else
    fprintf(stdout, "%uMb/s\n", link_usettings->base.speed);

    fprintf(stdout, "Duplex: ");
    switch (link_usettings->base.duplex) {
    case DUPLEX_HALF:
    fprintf(stdout, "Half\n");
    break;
    case DUPLEX_FULL:
    fprintf(stdout, "Full\n");
    break;
    default:
    fprintf(stdout, "Unknown! (%i)\n", link_usettings->base.duplex);
    break;
    };

    fprintf(stdout, "Port: ");
    switch (link_usettings->base.port) {
    case PORT_TP:
    fprintf(stdout, "Twisted Pair\n");
    break;
    case PORT_AUI:
    fprintf(stdout, "AUI\n");
    break;
    case PORT_BNC:
    fprintf(stdout, "BNC\n");
    break;
    case PORT_MII:
    fprintf(stdout, "MII\n");
    break;
    case PORT_FIBRE:
    fprintf(stdout, "FIBRE\n");
    break;
    case PORT_DA:
    fprintf(stdout, "Direct Attach Copper\n");
    break;

```

```
case PORT_NONE:
fprintf(stdout, "None\n");
break;
case PORT_OTHER:
fprintf(stdout, "Other\n");
break;
default:
fprintf(stdout, "Unknown! (%i)\n", link_usettings->base.port);
break;
};

fprintf(stdout, "PHYAD: %d\n", link_usettings->base.phy_address);
fprintf(stdout, "Transceiver: ");
switch (link_usettings->deprecated.transceiver) {
case XCVR_INTERNAL:
fprintf(stdout, "internal\n");
break;
case XCVR_EXTERNAL:
fprintf(stdout, "external\n");
break;
default:
fprintf(stdout, "Unknown!\n");
break;
};

fprintf(stdout, "Auto-negotiation: %s\n",
(link_usettings->base.autoneg == AUTONEG_DISABLE) ?
"off" : "on");

if (link_usettings->base.port == PORT_TP) {
fprintf(stdout, "MDI-X: ");
if (link_usettings->base.eth_tp_mdix_ctrl == ETH_TP_MDI) {
fprintf(stdout, "off (forced)\n");
} else if (link_usettings->base.eth_tp_mdix_ctrl
== ETH_TP_MDI_X) {
fprintf(stdout, "on (forced)\n");
} else {
switch (link_usettings->base.eth_tp_mdix) {
case ETH_TP_MDI:
fprintf(stdout, "off");
break;
case ETH_TP_MDI_X:
fprintf(stdout, "on");
break;
default:
fprintf(stdout, "Unknown");
break;
}
}
if (link_usettings->base.eth_tp_mdix_ctrl
== ETH_TP_MDI_AUTO)
fprintf(stdout, " (auto)");
fprintf(stdout, "\n");
}
}
```

```

return 0;
}

static int dump_drvinfo(struct ethtool_drvinfo *info)
{
    fprintf(stdout,
        "driver: %s\n"
        "version: %s\n"
        "firmware-version: %s\n"
        "expansion-rom-version: %s\n"
        "bus-info: %s\n"
        "supports-statistics: %s\n"
        "supports-test: %s\n"
        "supports-EEPROM-access: %s\n"
        "supports-register-dump: %s\n"
        "supports-priv-flags: %s\n",
        (int)sizeof(info->driver), info->driver,
        (int)sizeof(info->version), info->version,
        (int)sizeof(info->fw_version), info->fw_version,
        (int)sizeof(info->erom_version), info->erom_version,
        (int)sizeof(info->bus_info), info->bus_info,
        info->n_stats ? "yes" : "no",
        info->testinfo_len ? "yes" : "no",
        info->eedump_len ? "yes" : "no",
        info->regdump_len ? "yes" : "no",
        info->n_priv_flags ? "yes" : "no");

    return 0;
}

static int parse_wolopts(char *optstr, u32 *data)
{
    *data = 0;
    while (*optstr) {
        switch (*optstr) {
            case 'p':
                *data |= WAKE_PHY;
                break;
            case 'u':
                *data |= WAKE_UCAST;
                break;
            case 'm':
                *data |= WAKE_MCAST;
                break;
            case 'b':
                *data |= WAKE_BCAST;
                break;
            case 'a':
                *data |= WAKE_ARP;
                break;
            case 'g':
                *data |= WAKE_MAGIC;
                break;
            case 's':

```

```
*data |= WAKE_MAGICSECURE;
break;
case 'd':
    *data = 0;
break;
default:
    return -1;
}
optstr++;
}
return 0;
}

static char *unparse_wolopts(int wolopts)
{
    static char buf[16];
    char *p = buf;

    memset(buf, 0, sizeof(buf));

    if (wolopts) {
        if (wolopts & WAKE_PHY)
            *p++ = 'p';
        if (wolopts & WAKE_UCAST)
            *p++ = 'u';
        if (wolopts & WAKE_MCAST)
            *p++ = 'm';
        if (wolopts & WAKE_BCAST)
            *p++ = 'b';
        if (wolopts & WAKE_ARP)
            *p++ = 'a';
        if (wolopts & WAKE_MAGIC)
            *p++ = 'g';
        if (wolopts & WAKE_MAGICSECURE)
            *p++ = 's';
        } else {
            *p = 'd';
        }

    return buf;
}

static int dump_wol(struct ethtool_wolinfo *wol)
{
    {
        fprintf(stdout, "Supports Wake-on: %s\n",
            unparse_wolopts(wol->supported));
        fprintf(stdout, "Wake-on: %s\n",
            unparse_wolopts(wol->wolopts));
        if (wol->supported & WAKE_MAGICSECURE) {
            int i;
            int delim = 0;
            fprintf(stdout, "    SecureOn password: ");
            for (i = 0; i < SOPASS_MAX; i++) {
                fprintf(stdout, "%s%02x", delim?" ":"", wol->sopass[i]);
            }
        }
    }
}
```

```

    delim=1;
    }
    fprintf(stdout, "\n");
    }

    return 0;
}

static int parse_rxhashopts(char *optstr, u32 *data)
{
    *data = 0;
    while (*optstr) {
        switch (*optstr) {
            case 'm':
                *data |= RXH_L2DA;
                break;
            case 'v':
                *data |= RXH_VLAN;
                break;
            case 't':
                *data |= RXH_L3_PROTO;
                break;
            case 's':
                *data |= RXH_IP_SRC;
                break;
            case 'd':
                *data |= RXH_IP_DST;
                break;
            case 'f':
                *data |= RXH_L4_B_0_1;
                break;
            case 'n':
                *data |= RXH_L4_B_2_3;
                break;
            case 'r':
                *data |= RXH_DISCARD;
                break;
            default:
                return -1;
        }
        optstr++;
    }
    return 0;
}

static char *unparse_rxhashopts(u64 opts)
{
    {
        static char buf[300];

        memset(buf, 0, sizeof(buf));

        if (opts) {
            if (opts & RXH_L2DA) {
                strcat(buf, "L2DA\n");
            }

```

```

    if (opts & RXH_VLAN) {
        strcat(buf, "VLAN tag\n");
    }
    if (opts & RXH_L3_PROTO) {
        strcat(buf, "L3 proto\n");
    }
    if (opts & RXH_IP_SRC) {
        strcat(buf, "IP SA\n");
    }
    if (opts & RXH_IP_DST) {
        strcat(buf, "IP DA\n");
    }
    if (opts & RXH_L4_B_0_1) {
        strcat(buf, "L4 bytes 0 & 1 [TCP/UDP src port]\n");
    }
    if (opts & RXH_L4_B_2_3) {
        strcat(buf, "L4 bytes 2 & 3 [TCP/UDP dst port]\n");
    }
    } else {
        sprintf(buf, "None");
    }

    return buf;
}

static int convert_string_to_hashkey(char *rss_hkey, u32 key_size,
    const char *rss_hkey_string)
{
    {
        u32 i = 0;
        int hex_byte, len;

        do {
            if (i > (key_size - 1)) {
                fprintf(stderr,
                    "Key is too long for device (%u > %u)\n",
                    i + 1, key_size);
                goto err;
            }

            if (sscanf(rss_hkey_string, "%2x%n", &hex_byte, &len) < 1 ||
                len != 2) {
                fprintf(stderr, "Invalid RSS hash key format\n");
                goto err;
            }

            rss_hkey[i++] = hex_byte;
            rss_hkey_string += 2;

            if (*rss_hkey_string == ':') {
                rss_hkey_string++;
            } else if (*rss_hkey_string != '\0') {
                fprintf(stderr, "Invalid RSS hash key format\n");
                goto err;
            }

```

```

    } while (*rss_hkey_string);

    if (i != key_size) {
        fprintf(stderr, "Key is too short for device (%u < %u)\n",
            i, key_size);
        goto err;
    }

    return 0;
err:
    return 2;
}

static int parse_hkey(char **rss_hkey, u32 key_size,
    const char *rss_hkey_string)
{
    if (!key_size) {
        fprintf(stderr,
            "Cannot set RX flow hash configuration:\n"
            " Hash key setting not supported\n");
        return 1;
    }

    *rss_hkey = malloc(key_size);
    if (!(*rss_hkey)) {
        perror("Cannot allocate memory for RSS hash key");
        return 1;
    }

    if (convert_string_to_hashkey(*rss_hkey, key_size,
        rss_hkey_string)) {
        free(*rss_hkey);
        *rss_hkey = NULL;
        return 2;
    }
    return 0;
}

static const struct {
    const char *name;
    int (*func)(struct ethtool_drvinfo *info, struct ethtool_regs *regs);

} driver_list[] = {
#ifdef ETHTOOL_ENABLE_PRETTY_DUMP
    { "8139cp", realtek_dump_regs },
    { "8139too", realtek_dump_regs },
    { "r8169", realtek_dump_regs },
    { "de2104x", de2104x_dump_regs },
    { "e1000", e1000_dump_regs },
    { "e1000e", e1000_dump_regs },
    { "igb", igb_dump_regs },
    { "ixgb", ixgb_dump_regs },
    { "ixgbe", ixgbe_dump_regs },
    { "ixgbev", ixgbev_dump_regs },
    { "ixgbev", ixgbev_dump_regs },

```



```

{ "natsemi", natsemi_dump_regs },
{ "e100", e100_dump_regs },
{ "amd8111e", amd8111e_dump_regs },
{ "pcnet32", pcnet32_dump_regs },
{ "fec_8xx", fec_8xx_dump_regs },
{ "ibm_emac", ibm_emac_dump_regs },
{ "tg3", tg3_dump_regs },
{ "skge", skge_dump_regs },
{ "sky2", sky2_dump_regs },
    { "vioc", vioc_dump_regs },
    { "smc911x", smc911x_dump_regs },
    { "at76c50x-usb", at76c50x_usb_dump_regs },
    { "sfc", sfc_dump_regs },
{ "st_mac100", st_mac100_dump_regs },
{ "st_gmac", st_gmac_dump_regs },
{ "et131x", et131x_dump_regs },
{ "altera_tse", altera_tse_dump_regs },
{ "vmxnet3", vmxnet3_dump_regs },
#endif
};

void dump_hex(FILE *file, const u8 *data, int len, int offset)
{
    int i;

    fprintf(file, "Offset\t\tValues\n");
    fprintf(file, "-----\t\t-----");
    for (i = 0; i < len; i++) {
        if (i % 16 == 0)
            fprintf(file, "\n0x%04x:\t\t", i + offset);
        fprintf(file, "%02x ", data[i]);
    }
    fprintf(file, "\n");
}

static int dump_regs(int gregs_dump_raw, int gregs_dump_hex,
    struct ethtool_drvinfo *info, struct ethtool_regs *regs)
{
    int i;

    if (gregs_dump_raw) {
        fwrite(regs->data, regs->len, 1, stdout);
        return 0;
    }

    if (!gregs_dump_hex)
        for (i = 0; i < ARRAY_SIZE(driver_list); i++)
            if (!strcmp(driver_list[i].name, info->driver,
                ETHTOOL_BUSINFO_LEN)) {
                if (driver_list[i].func(info, regs) == 0)
                    return 0;
                /* This version (or some other
                 * variation in the dump format) is
                 * not handled; fall back to hex

```

```

    */
    break;
}

dump_hex(stdout, regs->data, regs->len, 0);

return 0;
}

static int dump_eeprom(int geeprom_dump_raw, struct ethtool_drvinfo *info,
    struct ethtool_eeprom *ee)
{
    if (geeprom_dump_raw) {
        fwrite(ee->data, 1, ee->len, stdout);
        return 0;
    }
    #ifndef ETHTOOL_ENABLE_PRETTY_DUMP
    if (!strcmp("natsemi", info->driver, ETHTOOL_BUSINFO_LEN)) {
        return natsemi_dump_eeprom(info, ee);
    } else if (!strcmp("tg3", info->driver, ETHTOOL_BUSINFO_LEN)) {
        return tg3_dump_eeprom(info, ee);
    }
    #endif
    dump_hex(stdout, ee->data, ee->len, ee->offset);

    return 0;
}

static int dump_test(struct ethtool_test *test,
    struct ethtool_gstrings *strings)
{
    int i, rc;

    rc = test->flags & ETH_TEST_FL_FAILED;
    fprintf(stdout, "The test result is %s\n", rc ? "FAIL" : "PASS");

    if (test->flags & ETH_TEST_FL_EXTERNAL_LB)
        fprintf(stdout, "External loopback test was %sexecuted\n",
            (test->flags & ETH_TEST_FL_EXTERNAL_LB_DONE) ?
            "" : "not ");

    if (strings->len)
        fprintf(stdout, "The test extra info:\n");

    for (i = 0; i < strings->len; i++) {
        fprintf(stdout, "%s\t %d\n",
            (char *) (strings->data + i * ETH_GSTRING_LEN),
            (u32) test->data[i]);
    }

    fprintf(stdout, "\n");
    return rc;
}

static int dump_pause(const struct ethtool_pauseparam *epause,

```

```
    u32 advertising, u32 lp_advertising)
{
    fprintf(stdout,
    "Autonegotiate:%s\n"
    "RX:%s\n"
    "TX:%s\n",
    epause->autoneg ? "on" : "off",
    epause->rx_pause ? "on" : "off",
    epause->tx_pause ? "on" : "off");

    if (lp_advertising) {
        int an_rx = 0, an_tx = 0;

        /* Work out negotiated pause frame usage per
         * IEEE 802.3-2005 table 28B-3.
         */
        if (advertising & lp_advertising & ADVERTISED_Pause) {
            an_tx = 1;
            an_rx = 1;
        } else if (advertising & lp_advertising &
            ADVERTISED_Asym_Pause) {
            if (advertising & ADVERTISED_Pause)
                an_rx = 1;
            else if (lp_advertising & ADVERTISED_Pause)
                an_tx = 1;
        }

        fprintf(stdout,
        "RX negotiated:%s\n"
        "TX negotiated:%s\n",
        an_rx ? "on" : "off",
        an_tx ? "on" : "off");
    }

    fprintf(stdout, "\n");
    return 0;
}

static int dump_ring(const struct ethtool_ringparam *ering)
{
    fprintf(stdout,
    "Pre-set maximums:\n"
    "RX:%u\n"
    "RX Mini:%u\n"
    "RX Jumbo:%u\n"
    "TX:%u\n",
    ering->rx_max_pending,
    ering->rx_mini_max_pending,
    ering->rx_jumbo_max_pending,
    ering->tx_max_pending);

    fprintf(stdout,
    "Current hardware settings:\n"
    "RX:%u\n"
```

```

"RX Mini:%u\n"
"RX Jumbo:%u\n"
"TX:%u\n",
ering->rx_pending,
ering->rx_mini_pending,
ering->rx_jumbo_pending,
ering->tx_pending);

fprintf(stdout, "\n");
return 0;
}

static int dump_channels(const struct ethtool_channels *echannels)
{
fprintf(stdout,
"Pre-set maximums:\n"
"RX:%u\n"
"TX:%u\n"
"Other:%u\n"
"Combined:%u\n",
echannels->max_rx, echannels->max_tx,
echannels->max_other,
echannels->max_combined);

fprintf(stdout,
"Current hardware settings:\n"
"RX:%u\n"
"TX:%u\n"
"Other:%u\n"
"Combined:%u\n",
echannels->rx_count, echannels->tx_count,
echannels->other_count,
echannels->combined_count);

fprintf(stdout, "\n");
return 0;
}

static int dump_coalesce(const struct ethtool_coalesce *ecoal)
{
fprintf(stdout, "Adaptive RX: %s TX: %s\n",
ecoal->use_adaptive_rx_coalesce ? "on" : "off",
ecoal->use_adaptive_tx_coalesce ? "on" : "off");

fprintf(stdout,
"stats-block-usecs: %u\n"
"sample-interval: %u\n"
"pkt-rate-low: %u\n"
"pkt-rate-high: %u\n"
"\n"
"rx-usecs: %u\n"
"rx-frames: %u\n"
"rx-usecs-irq: %u\n"
"rx-frames-irq: %u\n"
"\n")

```

```

"tx-usecs: %u\n"
"tx-frames: %u\n"
"tx-usecs-irq: %u\n"
"tx-frames-irq: %u\n"
"\n"
"rx-usecs-low: %u\n"
"rx-frame-low: %u\n"
"tx-usecs-low: %u\n"
"tx-frame-low: %u\n"
"\n"
"rx-usecs-high: %u\n"
"rx-frame-high: %u\n"
"tx-usecs-high: %u\n"
"tx-frame-high: %u\n"
"\n",
ecoal->stats_block_coalesce_usecs,
ecoal->rate_sample_interval,
ecoal->pkt_rate_low,
ecoal->pkt_rate_high,

ecoal->rx_coalesce_usecs,
ecoal->rx_max_coalesced_frames,
ecoal->rx_coalesce_usecs_irq,
ecoal->rx_max_coalesced_frames_irq,

ecoal->tx_coalesce_usecs,
ecoal->tx_max_coalesced_frames,
ecoal->tx_coalesce_usecs_irq,
ecoal->tx_max_coalesced_frames_irq,

ecoal->rx_coalesce_usecs_low,
ecoal->rx_max_coalesced_frames_low,
ecoal->tx_coalesce_usecs_low,
ecoal->tx_max_coalesced_frames_low,

ecoal->rx_coalesce_usecs_high,
ecoal->rx_max_coalesced_frames_high,
ecoal->tx_coalesce_usecs_high,
ecoal->tx_max_coalesced_frames_high);

return 0;
}

struct feature_state {
u32 off_flags;
struct ethtool_gfeatures features;
};

static void dump_one_feature(const char *indent, const char *name,
    const struct feature_state *state,
    const struct feature_state *ref_state,
    u32 index)
{
if (ref_state &&

```

```

    !(FEATURE_BIT_IS_SET(state->features.features, index, active) ^
    FEATURE_BIT_IS_SET(ref_state->features.features, index, active)))
return;

printf("%s%s: %s%s\n",
    indent, name,
    FEATURE_BIT_IS_SET(state->features.features, index, active) ?
    "on" : "off",
    (!FEATURE_BIT_IS_SET(state->features.features, index, available)
    || FEATURE_BIT_IS_SET(state->features.features, index,
    never_changed))
    ? "[fixed]"
    : (FEATURE_BIT_IS_SET(state->features.features, index, requested)
    ^ FEATURE_BIT_IS_SET(state->features.features, index, active))
    ? (FEATURE_BIT_IS_SET(state->features.features, index, requested)
    ? "[requested on]" : "[requested off]")
    : "");
}

static int linux_version_code(void)
{
    struct utsname utsname;
    unsigned version, patchlevel, sublevel = 0;

    if (uname(&utsname))
        return -1;
    if (sscanf(utsname.release, "%u.%u.%u", &version, &patchlevel, &sublevel) < 2)
        return -1;
    return KERNEL_VERSION(version, patchlevel, sublevel);
}

static void dump_features(const struct feature_defs *defs,
    const struct feature_state *state,
    const struct feature_state *ref_state)
{
    int kernel_ver = linux_version_code();
    u32 value;
    int indent;
    int i, j;

    for (i = 0; i < ARRAY_SIZE(off_flag_def); i++) {
        /* Don't show features whose state is unknown on this
        * kernel version
        */
        if (defs->off_flag_matched[i] == 0 &&
            off_flag_def[i].get_cmd == 0 &&
            kernel_ver < off_flag_def[i].min_kernel_ver)
            continue;

        value = off_flag_def[i].value;

        /* If this offload flag matches exactly one generic
        * feature then it's redundant to show the flag and
        * feature states separately. Otherwise, show the
        * flag state first.

```

```

    */
    if (defs->off_flag_matched[i] != 1 &&
        (!ref_state ||
         (state->off_flags ^ ref_state->off_flags) & value)) {
    printf("%s: %s\n",
        off_flag_def[i].long_name,
        (state->off_flags & value) ? "on" : "off");
    indent = 1;
    } else {
    indent = 0;
    }

    /* Show matching features */
    for (j = 0; j < defs->n_features; j++) {
    if (defs->def[j].off_flag_index != i)
    continue;
    if (defs->off_flag_matched[i] != 1)
    /* Show all matching feature states */
    dump_one_feature(indent ? "\t" : "",
        defs->def[j].name,
        state, ref_state, j);
    else
    /* Show full state with the old flag name */
    dump_one_feature("", off_flag_def[i].long_name,
        state, ref_state, j);
    }
    }

    /* Show all unmatched features that have non-null names */
    for (j = 0; j < defs->n_features; j++)
    if (defs->def[j].off_flag_index < 0 && defs->def[j].name[0])
    dump_one_feature("", defs->def[j].name,
        state, ref_state, j);
    }

    static int dump_rxfhash(int fhash, u64 val)
    {
    switch (fhash) {
    case TCP_V4_FLOW:
    fprintf(stdout, "TCP over IPV4 flows");
    break;
    case UDP_V4_FLOW:
    fprintf(stdout, "UDP over IPV4 flows");
    break;
    case SCTP_V4_FLOW:
    fprintf(stdout, "SCTP over IPV4 flows");
    break;
    case AH_ESP_V4_FLOW:
    case AH_V4_FLOW:
    case ESP_V4_FLOW:
    fprintf(stdout, "IPSEC AH/ESP over IPV4 flows");
    break;
    case TCP_V6_FLOW:
    fprintf(stdout, "TCP over IPV6 flows");

```

```

break;
case UDP_V6_FLOW:
fprintf(stdout, "UDP over IPV6 flows");
break;
case SCTP_V6_FLOW:
fprintf(stdout, "SCTP over IPV6 flows");
break;
case AH_ESP_V6_FLOW:
case AH_V6_FLOW:
case ESP_V6_FLOW:
fprintf(stdout, "IPSEC AH/ESP over IPV6 flows");
break;
default:
break;
}

if (val & RXH_DISCARD) {
fprintf(stdout, " - All matching flows discarded on RX\n");
return 0;
}
fprintf(stdout, " use these fields for computing Hash flow key:\n");

fprintf(stdout, "%s\n", unparse_rxfhashopts(val));

return 0;
}

static void dump_eecmd(struct ethtool_eee *ep)
{
ETHTOOL_DECLARE_LINK_MODE_MASK(link_mode);

fprintf(stdout, "EEE status: ");
if (!ep->supported) {
fprintf(stdout, "not supported\n");
return;
} else if (!ep->eee_enabled) {
fprintf(stdout, "disabled\n");
} else {
fprintf(stdout, "enabled - ");
if (ep->eee_active)
fprintf(stdout, "active\n");
else
fprintf(stdout, "inactive\n");
}

fprintf(stdout, "Tx LPI:");
if (ep->tx_lpi_enabled)
fprintf(stdout, " %d (us)\n", ep->tx_lpi_timer);
else
fprintf(stdout, " disabled\n");

ethtool_link_mode_zero(link_mode);

link_mode[0] = ep->supported;
dump_link_caps("Supported EEE", "", link_mode, 1);

```



```

link_mode[0] = ep->advertised;
dump_link_caps("Advertised EEE", "", link_mode, 1);

link_mode[0] = ep->lp_advertised;
dump_link_caps("Link partner advertised EEE", "", link_mode, 1);
}

#define N_SOTS 7

static char *so_timestamping_labels[N_SOTS] = {
"hardware-transmit    (SOF_TIMESTAMPING_TX_HARDWARE)",
"software-transmit    (SOF_TIMESTAMPING_TX_SOFTWARE)",
"hardware-receive     (SOF_TIMESTAMPING_RX_HARDWARE)",
"software-receive     (SOF_TIMESTAMPING_RX_SOFTWARE)",
"software-system-clock (SOF_TIMESTAMPING_SOFTWARE)",
"hardware-legacy-clock (SOF_TIMESTAMPING_SYS_HARDWARE)",
"hardware-raw-clock   (SOF_TIMESTAMPING_RAW_HARDWARE)",
};

#define N_TX_TYPES (HWTSTAMP_TX_ONESTEP_SYNC + 1)

static char *tx_type_labels[N_TX_TYPES] = {
"off      (HWTSTAMP_TX_OFF)",
"on       (HWTSTAMP_TX_ON)",
"one-step-sync (HWTSTAMP_TX_ONESTEP_SYNC)",
};

#define N_RX_FILTERS (HWTSTAMP_FILTER_PTP_V2_DELAY_REQ + 1)

static char *rx_filter_labels[N_RX_FILTERS] = {
"none      (HWTSTAMP_FILTER_NONE)",
"all       (HWTSTAMP_FILTER_ALL)",
"some      (HWTSTAMP_FILTER_SOME)",
"ptpv1-l4-event  (HWTSTAMP_FILTER_PTP_V1_L4_EVENT)",
"ptpv1-l4-sync   (HWTSTAMP_FILTER_PTP_V1_L4_SYNC)",
"ptpv1-l4-delay-req (HWTSTAMP_FILTER_PTP_V1_L4_DELAY_REQ)",
"ptpv2-l4-event  (HWTSTAMP_FILTER_PTP_V2_L4_EVENT)",
"ptpv2-l4-sync   (HWTSTAMP_FILTER_PTP_V2_L4_SYNC)",
"ptpv2-l4-delay-req (HWTSTAMP_FILTER_PTP_V2_L4_DELAY_REQ)",
"ptpv2-l2-event  (HWTSTAMP_FILTER_PTP_V2_L2_EVENT)",
"ptpv2-l2-sync   (HWTSTAMP_FILTER_PTP_V2_L2_SYNC)",
"ptpv2-l2-delay-req (HWTSTAMP_FILTER_PTP_V2_L2_DELAY_REQ)",
"ptpv2-event     (HWTSTAMP_FILTER_PTP_V2_EVENT)",
"ptpv2-sync      (HWTSTAMP_FILTER_PTP_V2_SYNC)",
"ptpv2-delay-req (HWTSTAMP_FILTER_PTP_V2_DELAY_REQ)",
};

static int dump_tsinfo(const struct ethtool_ts_info *info)
{
int i;

fprintf(stdout, "Capabilities:\n");

```

```

for (i = 0; i < N_SOTS; i++) {
if (info->so_timestamping & (1 << i))
fprintf(stdout, "\t%s\n", so_timestamping_labels[i]);
}

fprintf(stdout, "PTP Hardware Clock: ");

if (info->phc_index < 0)
fprintf(stdout, "none\n");
else
fprintf(stdout, "%d\n", info->phc_index);

fprintf(stdout, "Hardware Transmit Timestamp Modes:");

if (!info->tx_types)
fprintf(stdout, " none\n");
else
fprintf(stdout, "\n");

for (i = 0; i < N_TX_TYPES; i++) {
if (info->tx_types & (1 << i))
fprintf(stdout, "\t%s\n", tx_type_labels[i]);
}

fprintf(stdout, "Hardware Receive Filter Modes:");

if (!info->rx_filters)
fprintf(stdout, " none\n");
else
fprintf(stdout, "\n");

for (i = 0; i < N_RX_FILTERS; i++) {
if (info->rx_filters & (1 << i))
fprintf(stdout, "\t%s\n", rx_filter_labels[i]);
}

return 0;
}

static struct ethtool_gstrings *
get_stringset(struct cmd_context *ctx, enum ethtool_stringset set_id,
               ptrdiff_t drvinfo_offset, int null_terminate)
{
    struct {
        struct ethtool_sset_info hdr;
        u32 buf[1];
    } sset_info;
    struct ethtool_drvinfo drvinfo;
    u32 len, i;
    struct ethtool_gstrings *strings;

    sset_info.hdr.cmd = ETHTOOL_GSSET_INFO;
    sset_info.hdr.reserved = 0;
    sset_info.hdr.sset_mask = 1ULL << set_id;
    if (send_ioctl(ctx, &sset_info) == 0) {

```

```

len = sset_info.hdr.sset_mask ? sset_info.hdr.data[0] : 0;
} else if (errno == EOPNOTSUPP && drvinfo_offset != 0) {
/* Fallback for old kernel versions */
drvinfo.cmd = ETHTOOL_GDRVINFO;
if (send_ioctl(ctx, &drvinfo))
return NULL;
len = *(u32 *)((char *)&drvinfo + drvinfo_offset);
} else {
return NULL;
}

strings = calloc(1, sizeof(*strings) + len * ETH_GSTRING_LEN);
if (!strings)
return NULL;

strings->cmd = ETHTOOL_GSTRINGS;
strings->string_set = set_id;
strings->len = len;
if (len != 0 && send_ioctl(ctx, strings)) {
free(strings);
return NULL;
}

if (null_terminate)
for (i = 0; i < len; i++)
strings->data[(i + 1) * ETH_GSTRING_LEN - 1] = 0;

return strings;
}

static struct feature_defs *get_feature_defs(struct cmd_context *ctx)
{
struct ethtool_gstrings *names;
struct feature_defs *defs;
u32 n_features;
int i, j;

names = get_stringset(ctx, ETH_SS_FEATURES, 0, 1);
if (names) {
n_features = names->len;
} else if (errno == EOPNOTSUPP || errno == EINVAL) {
/* Kernel doesn't support named features; not an error */
n_features = 0;
} else if (errno == EPERM) {
/* Kernel bug: ETHTOOL_GSSET_INFO was privileged.
 * Work around it. */
n_features = 0;
} else {
return NULL;
}

defs = malloc(sizeof(*defs) + sizeof(defs->def[0]) * n_features);
if (!defs)
return NULL;

```

```

defs->n_features = n_features;
memset(defs->off_flag_matched, 0, sizeof(defs->off_flag_matched));

/* Copy out feature names and find those associated with legacy flags */
for (i = 0; i < defs->n_features; i++) {
    memcpy(defs->def[i].name, names->data + i * ETH_GSTRING_LEN,
           ETH_GSTRING_LEN);
    defs->def[i].off_flag_index = -1;

    for (j = 0;
         j < ARRAY_SIZE(off_flag_def) &&
         defs->def[i].off_flag_index < 0;
         j++) {
        const char *pattern =
            off_flag_def[j].kernel_name;
        const char *name = defs->def[i].name;
        for (;) {
            if (*pattern == '*') {
                /* There is only one wildcard; so
                 * switch to a suffix comparison */
                size_t pattern_len =
                    strlen(pattern + 1);
                size_t name_len = strlen(name);
                if (name_len < pattern_len)
                    break; /* name is too short */
                name += name_len - pattern_len;
                ++pattern;
            } else if (*pattern != *name) {
                break; /* mismatch */
            } else if (*pattern == 0) {
                defs->def[i].off_flag_index = j;
                defs->off_flag_matched[j]++;
                break;
            } else {
                ++name;
                ++pattern;
            }
        }
    }

    free(names);
    return defs;
}

static int do_gdrv(struct cmd_context *ctx)
{
    int err;
    struct ethtool_drvinfo drvinfo;

    if (ctx->argc != 0)
        exit_bad_args();

    drvinfo.cmd = ETHTOOL_GDRVINFO;

```

```
err = send_ioctl(ctx, &drvinfo);
if (err < 0) {
    perror("Cannot get driver information");
    return 71;
}
return dump_drvinfo(&drvinfo);
}

static int do_gpause(struct cmd_context *ctx)
{
    struct ethtool_pauseparam epause;
    struct ethtool_cmd ecmd;
    int err;

    if (ctx->argc != 0)
        exit_bad_args();

    fprintf(stdout, "Pause parameters for %s:\n", ctx->devname);

    epause.cmd = ETHTOOL_GPAUSEPARAM;
    err = send_ioctl(ctx, &epause);
    if (err) {
        perror("Cannot get device pause settings");
        return 76;
    }

    if (epause.autoneg) {
        ecmd.cmd = ETHTOOL_GSET;
        err = send_ioctl(ctx, &ecmd);
        if (err) {
            perror("Cannot get device settings");
            return 1;
        }
        dump_pause(&epause, ecmd.advertising, ecmd.lp_advertising);
    } else {
        dump_pause(&epause, 0, 0);
    }

    return 0;
}

static void do_generic_set1(struct cmdline_info *info, int *changed_out)
{
    int wanted, *v1, *v2;

    v1 = info->wanted_val;
    wanted = *v1;

    if (wanted < 0)
        return;

    v2 = info->ioctl_val;
    if (wanted == *v2) {
        fprintf(stderr, "%s unmodified, ignoring\n", info->name);
    }
}
```

```

    } else {
        *v2 = wanted;
        *changed_out = 1;
    }
}

static void do_generic_set(struct cmdline_info *info,
    unsigned int n_info,
    int *changed_out)
{
    unsigned int i;

    for (i = 0; i < n_info; i++)
        do_generic_set1(&info[i], changed_out);
}

static int do_spause(struct cmd_context *ctx)
{
    struct ethtool_pauseparam epause;
    int gpause_changed = 0;
    int pause_autoneg_wanted = -1;
    int pause_rx_wanted = -1;
    int pause_tx_wanted = -1;
    struct cmdline_info cmdline_pause[] = {
        { "autoneg", CMDL_BOOL, &pause_autoneg_wanted,
          &epause.autoneg },
        { "rx", CMDL_BOOL, &pause_rx_wanted, &epause.rx_pause },
        { "tx", CMDL_BOOL, &pause_tx_wanted, &epause.tx_pause },
    };
    int err, changed = 0;

    parse_generic_cmdline(ctx, &gpause_changed,
        cmdline_pause, ARRAY_SIZE(cmdline_pause));

    epause.cmd = ETHTOOL_GPAUSEPARAM;
    err = send_ioctl(ctx, &epause);
    if (err) {
        perror("Cannot get device pause settings");
        return 77;
    }

    do_generic_set(cmdline_pause, ARRAY_SIZE(cmdline_pause), &changed);

    if (!changed) {
        fprintf(stderr, "no pause parameters changed, aborting\n");
        return 78;
    }

    epause.cmd = ETHTOOL_SPAUSEPARAM;
    err = send_ioctl(ctx, &epause);
    if (err) {
        perror("Cannot set device pause parameters");
        return 79;
    }
}

```

```
return 0;
}

static int do_sring(struct cmd_context *ctx)
{
    struct ethtool_ringparam ering;
    int gring_changed = 0;
    s32 ring_rx_wanted = -1;
    s32 ring_rx_mini_wanted = -1;
    s32 ring_rx_jumbo_wanted = -1;
    s32 ring_tx_wanted = -1;
    struct cmdline_info cmdline_ring[] = {
        { "rx", CMDL_S32, &ring_rx_wanted, &ering.rx_pending },
        { "rx-mini", CMDL_S32, &ring_rx_mini_wanted,
          &ering.rx_mini_pending },
        { "rx-jumbo", CMDL_S32, &ring_rx_jumbo_wanted,
          &ering.rx_jumbo_pending },
        { "tx", CMDL_S32, &ring_tx_wanted, &ering.tx_pending },
    };
    int err, changed = 0;

    parse_generic_cmdline(ctx, &gring_changed,
        cmdline_ring, ARRAY_SIZE(cmdline_ring));

    ering.cmd = ETHTOOL_GRINGPARAM;
    err = send_ioctl(ctx, &ering);
    if (err) {
        perror("Cannot get device ring settings");
        return 76;
    }

    do_generic_set(cmdline_ring, ARRAY_SIZE(cmdline_ring), &changed);

    if (!changed) {
        fprintf(stderr, "no ring parameters changed, aborting\n");
        return 80;
    }

    ering.cmd = ETHTOOL_SRINGPARAM;
    err = send_ioctl(ctx, &ering);
    if (err) {
        perror("Cannot set device ring parameters");
        return 81;
    }

    return 0;
}

static int do_gring(struct cmd_context *ctx)
{
    struct ethtool_ringparam ering;
    int err;

    if (ctx->argc != 0)
```

```

exit_bad_args();

fprintf(stdout, "Ring parameters for %s:\n", ctx->devname);

ering.cmd = ETHTOOL_GRINGPARAM;
err = send_ioctl(ctx, &ering);
if (err == 0) {
    err = dump_ring(&ering);
    if (err)
        return err;
} else {
    perror("Cannot get device ring settings");
    return 76;
}

return 0;
}

static int do_schannels(struct cmd_context *ctx)
{
    struct ethtool_channels echannels;
    int gchannels_changed;
    s32 channels_rx_wanted = -1;
    s32 channels_tx_wanted = -1;
    s32 channels_other_wanted = -1;
    s32 channels_combined_wanted = -1;
    struct cmdline_info cmdline_channels[] = {
        { "rx", CMDL_S32, &channels_rx_wanted, &echannels.rx_count },
        { "tx", CMDL_S32, &channels_tx_wanted, &echannels.tx_count },
        { "other", CMDL_S32, &channels_other_wanted,
          &echannels.other_count },
        { "combined", CMDL_S32, &channels_combined_wanted,
          &echannels.combined_count },
    };
    int err, changed = 0;

    parse_generic_cmdline(ctx, &gchannels_changed,
        cmdline_channels, ARRAY_SIZE(cmdline_channels));

    echannels.cmd = ETHTOOL_GCHANNELS;
    err = send_ioctl(ctx, &echannels);
    if (err) {
        perror("Cannot get device channel parameters");
        return 1;
    }

    do_generic_set(cmdline_channels, ARRAY_SIZE(cmdline_channels),
        &changed);

    if (!changed) {
        fprintf(stderr, "no channel parameters changed, aborting\n");
        fprintf(stderr, "current values: tx %u rx %u other %u"
            " combined %u\n", echannels.rx_count,
            echannels.tx_count, echannels.other_count,
            echannels.combined_count);
    }
}

```



```
return 1;
}

echannels.cmd = ETHTOOL_SCHANNELS;
err = send_ioctl(ctx, &echannels);
if (err) {
    perror("Cannot set device channel parameters");
    return 1;
}

return 0;
}

static int do_gchannels(struct cmd_context *ctx)
{
    struct ethtool_channels echannels;
    int err;

    if (ctx->argc != 0)
        exit_bad_args();

    fprintf(stdout, "Channel parameters for %s:\n", ctx->devname);

    echannels.cmd = ETHTOOL_GCHANNELS;
    err = send_ioctl(ctx, &echannels);
    if (err == 0) {
        err = dump_channels(&echannels);
        if (err)
            return err;
    } else {
        perror("Cannot get device channel parameters\n");
        return 1;
    }
    return 0;
}

static int do_gcoalesce(struct cmd_context *ctx)
{
    struct ethtool_coalesce ecoal;
    int err;

    if (ctx->argc != 0)
        exit_bad_args();

    fprintf(stdout, "Coalesce parameters for %s:\n", ctx->devname);

    ecoal.cmd = ETHTOOL_GCOALESCE;
    err = send_ioctl(ctx, &ecoa);
    if (err == 0) {
        err = dump_coalesce(&ecoa);
        if (err)
            return err;
    } else {
```

```

perror("Cannot get device coalesce settings");
return 82;
}

return 0;
}

static int do_scoalesce(struct cmd_context *ctx)
{
    struct ethtool_coalesce ecoal;
    int gcoalesce_changed = 0;
    s32 coal_stats_wanted = -1;
    int coal_adaptive_rx_wanted = -1;
    int coal_adaptive_tx_wanted = -1;
    s32 coal_sample_rate_wanted = -1;
    s32 coal_pkt_rate_low_wanted = -1;
    s32 coal_pkt_rate_high_wanted = -1;
    s32 coal_rx_usec_wanted = -1;
    s32 coal_rx_frames_wanted = -1;
    s32 coal_rx_usec_irq_wanted = -1;
    s32 coal_rx_frames_irq_wanted = -1;
    s32 coal_tx_usec_wanted = -1;
    s32 coal_tx_frames_wanted = -1;
    s32 coal_tx_usec_irq_wanted = -1;
    s32 coal_tx_frames_irq_wanted = -1;
    s32 coal_rx_usec_low_wanted = -1;
    s32 coal_rx_frames_low_wanted = -1;
    s32 coal_tx_usec_low_wanted = -1;
    s32 coal_tx_frames_low_wanted = -1;
    s32 coal_rx_usec_high_wanted = -1;
    s32 coal_rx_frames_high_wanted = -1;
    s32 coal_tx_usec_high_wanted = -1;
    s32 coal_tx_frames_high_wanted = -1;
    struct cmdline_info cmdline_coalesce[] = {
        { "adaptive-rx", CMDL_BOOL, &coal_adaptive_rx_wanted,
          &ecoal.use_adaptive_rx_coalesce },
        { "adaptive-tx", CMDL_BOOL, &coal_adaptive_tx_wanted,
          &ecoal.use_adaptive_tx_coalesce },
        { "sample-interval", CMDL_S32, &coal_sample_rate_wanted,
          &ecoal.rate_sample_interval },
        { "stats-block-usecs", CMDL_S32, &coal_stats_wanted,
          &ecoal.stats_block_coalesce_usecs },
        { "pkt-rate-low", CMDL_S32, &coal_pkt_rate_low_wanted,
          &ecoal.pkt_rate_low },
        { "pkt-rate-high", CMDL_S32, &coal_pkt_rate_high_wanted,
          &ecoal.pkt_rate_high },
        { "rx-usecs", CMDL_S32, &coal_rx_usec_wanted,
          &ecoal.rx_coalesce_usecs },
        { "rx-frames", CMDL_S32, &coal_rx_frames_wanted,
          &ecoal.rx_max_coalesced_frames },
        { "rx-usecs-irq", CMDL_S32, &coal_rx_usec_irq_wanted,
          &ecoal.rx_coalesce_usecs_irq },
        { "rx-frames-irq", CMDL_S32, &coal_rx_frames_irq_wanted,
          &ecoal.rx_max_coalesced_frames_irq },
        { "tx-usecs", CMDL_S32, &coal_tx_usec_wanted,

```

```

    &ecoal.tx_coalesce_usecs },
    { "tx-frames", CMDL_S32, &coal_tx_frames_wanted,
      &ecoal.tx_max_coalesced_frames },
    { "tx-usecs-irq", CMDL_S32, &coal_tx_usec_irq_wanted,
      &ecoal.tx_coalesce_usecs_irq },
    { "tx-frames-irq", CMDL_S32, &coal_tx_frames_irq_wanted,
      &ecoal.tx_max_coalesced_frames_irq },
    { "rx-usecs-low", CMDL_S32, &coal_rx_usec_low_wanted,
      &ecoal.rx_coalesce_usecs_low },
    { "rx-frames-low", CMDL_S32, &coal_rx_frames_low_wanted,
      &ecoal.rx_max_coalesced_frames_low },
    { "tx-usecs-low", CMDL_S32, &coal_tx_usec_low_wanted,
      &ecoal.tx_coalesce_usecs_low },
    { "tx-frames-low", CMDL_S32, &coal_tx_frames_low_wanted,
      &ecoal.tx_max_coalesced_frames_low },
    { "rx-usecs-high", CMDL_S32, &coal_rx_usec_high_wanted,
      &ecoal.rx_coalesce_usecs_high },
    { "rx-frames-high", CMDL_S32, &coal_rx_frames_high_wanted,
      &ecoal.rx_max_coalesced_frames_high },
    { "tx-usecs-high", CMDL_S32, &coal_tx_usec_high_wanted,
      &ecoal.tx_coalesce_usecs_high },
    { "tx-frames-high", CMDL_S32, &coal_tx_frames_high_wanted,
      &ecoal.tx_max_coalesced_frames_high },
    };
    int err, changed = 0;

    parse_generic_cmdline(ctx, &gcoalesce_changed,
        cmdline_coalesce, ARRAY_SIZE(cmdline_coalesce));

    ecoal.cmd = ETHTOOL_GCOALESCE;
    err = send_ioctl(ctx, &ecoal);
    if (err) {
        perror("Cannot get device coalesce settings");
        return 76;
    }

    do_generic_set(cmdline_coalesce, ARRAY_SIZE(cmdline_coalesce),
        &changed);

    if (!changed) {
        fprintf(stderr, "no coalesce parameters changed, aborting\n");
        return 80;
    }

    ecoal.cmd = ETHTOOL_SCOALESCE;
    err = send_ioctl(ctx, &ecoal);
    if (err) {
        perror("Cannot set device coalesce parameters");
        return 81;
    }

    return 0;
}

```

```

static struct feature_state *
get_features(struct cmd_context *ctx, const struct feature_defs *defs)
{
    struct feature_state *state;
    struct ethtool_value eval;
    int err, allfail = 1;
    u32 value;
    int i;

    state = malloc(sizeof(*state) +
        FEATURE_BITS_TO_BLOCKS(defs->n_features) *
        sizeof(state->features.features[0]));
    if (!state)
        return NULL;

    state->off_flags = 0;

    for (i = 0; i < ARRAY_SIZE(off_flag_def); i++) {
        value = off_flag_def[i].value;
        if (!off_flag_def[i].get_cmd)
            continue;
        eval.cmd = off_flag_def[i].get_cmd;
        err = send_ioctl(ctx, &eval);
        if (err) {
            fprintf(stderr,
                "Cannot get device %s settings: %m\n",
                off_flag_def[i].long_name);
        } else {
            if (eval.data)
                state->off_flags |= value;
            allfail = 0;
        }
    }

    eval.cmd = ETHTOOL_GFLAGS;
    err = send_ioctl(ctx, &eval);
    if (err) {
        perror("Cannot get device flags");
    } else {
        state->off_flags |= eval.data & ETH_FLAG_EXT_MASK;
        allfail = 0;
    }

    if (defs->n_features) {
        state->features.cmd = ETHTOOL_GFEATURES;
        state->features.size = FEATURE_BITS_TO_BLOCKS(defs->n_features);
        err = send_ioctl(ctx, &state->features);
        if (err)
            perror("Cannot get device generic features");
        else
            allfail = 0;
    }

    if (allfail) {
        free(state);
    }
}

```

```

return NULL;
}

return state;
}

static int do_gfeatures(struct cmd_context *ctx)
{
    struct feature_defs *defs;
    struct feature_state *features;

    if (ctx->argc != 0)
        exit_bad_args();

    defs = get_feature_defs(ctx);
    if (!defs) {
        perror("Cannot get device feature names");
        return 1;
    }

    fprintf(stdout, "Features for %s:\n", ctx->devname);

    features = get_features(ctx, defs);
    if (!features) {
        fprintf(stdout, "no feature info available\n");
        return 1;
    }

    dump_features(defs, features, NULL);
    return 0;
}

static int do_sfeatures(struct cmd_context *ctx)
{
    struct feature_defs *defs;
    int any_changed = 0, any_mismatch = 0;
    u32 off_flags_wanted = 0;
    u32 off_flags_mask = 0;
    struct ethtool_sfeatures *efeatures;
    struct cmdline_info *cmdline_features;
    struct feature_state *old_state, *new_state;
    struct ethtool_value eval;
    int err;
    int i, j;

    defs = get_feature_defs(ctx);
    if (!defs) {
        perror("Cannot get device feature names");
        return 1;
    }
    if (defs->n_features) {
        efeatures = malloc(sizeof(*efeatures) +
            FEATURE_BITS_TO_BLOCKS(defs->n_features) *
            sizeof(efeatures->features[0]));
    }

```

```

if (!features) {
    perror("Cannot parse arguments");
    return 1;
}
efeatures->cmd = ETHTOOL_SFEATURES;
efeatures->size = FEATURE_BITS_TO_BLOCKS(defs->n_features);
memset(efeatures->features, 0,
        FEATURE_BITS_TO_BLOCKS(defs->n_features) *
        sizeof(efeatures->features[0]));
} else {
    efeatures = NULL;
}

/* Generate cmdline_info for legacy flags and kernel-named
 * features, and parse our arguments.
 */
cmdline_features = calloc(ARRAY_SIZE(off_flag_def) + defs->n_features,
    sizeof(cmdline_features[0]));
if (!cmdline_features) {
    perror("Cannot parse arguments");
    return 1;
}
for (i = 0; i < ARRAY_SIZE(off_flag_def); i++)
    flag_to_cmdline_info(off_flag_def[i].short_name,
        off_flag_def[i].value,
        &off_flags_wanted, &off_flags_mask,
        &cmdline_features[i]);
for (i = 0; i < defs->n_features; i++)
    flag_to_cmdline_info(
        defs->def[i].name, FEATURE_FIELD_FLAG(i),
        &FEATURE_WORD(efeatures->features, i, requested),
        &FEATURE_WORD(efeatures->features, i, valid),
        &cmdline_features[ARRAY_SIZE(off_flag_def) + i]);
parse_generic_cmdline(ctx, &any_changed, cmdline_features,
    ARRAY_SIZE(off_flag_def) + defs->n_features);
free(cmdline_features);

if (!any_changed) {
    fprintf(stdout, "no features changed\n");
    return 0;
}

old_state = get_features(ctx, defs);
if (!old_state)
    return 1;

if (efeatures) {
    /* For each offload that the user specified, update any
     * related features that the user did not specify and that
     * are not fixed. Warn if all related features are fixed.
     */
    for (i = 0; i < ARRAY_SIZE(off_flag_def); i++) {
        int fixed = 1;

        if (!(off_flags_mask & off_flag_def[i].value))

```

```

continue;

for (j = 0; j < defs->n_features; j++) {
if (defs->def[j].off_flag_index != i ||
    !FEATURE_BIT_IS_SET(
        old_state->features.features,
        j, available) ||
    FEATURE_BIT_IS_SET(
        old_state->features.features,
        j, never_changed))
continue;

fixed = 0;
if (!FEATURE_BIT_IS_SET(efeatures->features,
    j, valid)) {
FEATURE_BIT_SET(efeatures->features,
    j, valid);
if (off_flags_wanted &
    off_flag_def[i].value)
FEATURE_BIT_SET(
    efeatures->features,
    j, requested);
}
}

if (fixed)
fprintf(stderr, "Cannot change %s\n",
    off_flag_def[i].long_name);
}

err = send_ioctl(ctx, efeatures);
if (err < 0) {
perror("Cannot set device feature settings");
return 1;
}
} else {
for (i = 0; i < ARRAY_SIZE(off_flag_def); i++) {
if (!off_flag_def[i].set_cmd)
continue;
if (off_flags_mask & off_flag_def[i].value) {
eval.cmd = off_flag_def[i].set_cmd;
eval.data = !(off_flags_wanted &
    off_flag_def[i].value);
err = send_ioctl(ctx, &eval);
if (err) {
fprintf(stderr,
    "Cannot set device %s settings: %m\n",
    off_flag_def[i].long_name);
return 1;
}
}
}
}

if (off_flags_mask & ETH_FLAG_EXT_MASK) {

```

```

eval.cmd = ETHTOOL_SFLAGS;
eval.data = (old_state->off_flags & ~off_flags_mask &
    ETH_FLAG_EXT_MASK);
eval.data |= off_flags_wanted & ETH_FLAG_EXT_MASK;

err = send_ioctl(ctx, &eval);
if (err) {
    perror("Cannot set device flag settings");
    return 92;
}
}
}

/* Compare new state with requested state */
new_state = get_features(ctx, defs);
if (!new_state)
    return 1;
any_changed = new_state->off_flags != old_state->off_flags;
any_mismatch = (new_state->off_flags !=
    ((old_state->off_flags & ~off_flags_mask) |
    off_flags_wanted));
for (i = 0; i < FEATURE_BITS_TO_BLOCKS(defs->n_features); i++) {
    if (new_state->features.features[i].active !=
        old_state->features.features[i].active)
        any_changed = 1;
    if (new_state->features.features[i].active !=
        ((old_state->features.features[i].active &
        ~efeatures->features[i].valid) |
        efeatures->features[i].requested))
        any_mismatch = 1;
}
if (any_mismatch) {
    if (!any_changed) {
        fprintf(stderr,
            "Could not change any device features\n");
        return 1;
    }
    printf("Actual changes:\n");
    dump_features(defs, new_state, old_state);
}

return 0;
}

static struct ethtool_link_usettings *
do_ioctl_glinksettings(struct cmd_context *ctx)
{
    int err;
    struct {
        struct ethtool_link_settings req;
        __u32 link_mode_data[3 * ETHTOOL_LINK_MODE_MASK_MAX_KERNEL_NU32];
    } ecmd;
    struct ethtool_link_usettings *link_usettings;
    unsigned int u32_offs;

```



```
/* Handshake with kernel to determine number of words for link
 * mode bitmaps. When requested number of bitmap words is not
 * the one expected by kernel, the latter returns the integer
 * opposite of what it is expecting. We request length 0 below
 * (aka. invalid bitmap length) to get this info.
 */
memset(&ecmd, 0, sizeof(ecmd));
ecmd.req.cmd = ETHTOOL_GLINKSETTINGS;
err = send_ioctl(ctx, &ecmd);
if (err < 0)
return NULL;

/* see above: we expect a strictly negative value from kernel.
 */
if (ecmd.req.link_mode_masks_nwords >= 0
    || ecmd.req.cmd != ETHTOOL_GLINKSETTINGS)
return NULL;

/* got the real ecmd.req.link_mode_masks_nwords,
 * now send the real request
 */
ecmd.req.cmd = ETHTOOL_GLINKSETTINGS;
ecmd.req.link_mode_masks_nwords = -ecmd.req.link_mode_masks_nwords;
err = send_ioctl(ctx, &ecmd);
if (err < 0)
return NULL;

if (ecmd.req.link_mode_masks_nwords <= 0
    || ecmd.req.cmd != ETHTOOL_GLINKSETTINGS)
return NULL;

/* Convert to usettings struct */
link_usettings = calloc(1, sizeof(*link_usettings));
if (link_usettings == NULL)
return NULL;

/* keep transceiver 0 */
memcpy(&link_usettings->base, &ecmd.req, sizeof(link_usettings->base));

/* copy link mode bitmaps */
u32_offs = 0;
memcpy(link_usettings->link_modes.supported,
    &ecmd.link_mode_data[u32_offs],
    4 * ecmd.req.link_mode_masks_nwords);

u32_offs += ecmd.req.link_mode_masks_nwords;
memcpy(link_usettings->link_modes.advertising,
    &ecmd.link_mode_data[u32_offs],
    4 * ecmd.req.link_mode_masks_nwords);

u32_offs += ecmd.req.link_mode_masks_nwords;
memcpy(link_usettings->link_modes.lp_advertising,
    &ecmd.link_mode_data[u32_offs],
    4 * ecmd.req.link_mode_masks_nwords);
```

```

return link_usettings;
}

static int
do_ioctl_slinksettings(struct cmd_context *ctx,
    const struct ethtool_link_usettings *link_usettings)
{
    struct {
        struct ethtool_link_settings req;
        __u32 link_mode_data[3 * ETHTOOL_LINK_MODE_MASK_MAX_KERNEL_NU32];
    } ecmd;
    unsigned int u32_offs;

    /* refuse to send ETHTOOL_SLINKSETTINGS ioctl if
     * link_usettings was retrieved with ETHTOOL_GSET
     */
    if (link_usettings->base.cmd != ETHTOOL_GLINKSETTINGS)
        return -1;

    /* refuse to send ETHTOOL_SLINKSETTINGS ioctl if deprecated fields
     * were set
     */
    if (link_usettings->deprecated.transceiver)
        return -1;

    if (link_usettings->base.link_mode_masks_nwords <= 0)
        return -1;

    memcpy(&ecmd.req, &link_usettings->base, sizeof(ecmd.req));
    ecmd.req.cmd = ETHTOOL_SLINKSETTINGS;

    /* copy link mode bitmaps */
    u32_offs = 0;
    memcpy(&ecmd.link_mode_data[u32_offs],
        link_usettings->link_modes.supported,
        4 * ecmd.req.link_mode_masks_nwords);

    u32_offs += ecmd.req.link_mode_masks_nwords;
    memcpy(&ecmd.link_mode_data[u32_offs],
        link_usettings->link_modes.advertising,
        4 * ecmd.req.link_mode_masks_nwords);

    u32_offs += ecmd.req.link_mode_masks_nwords;
    memcpy(&ecmd.link_mode_data[u32_offs],
        link_usettings->link_modes.lp_advertising,
        4 * ecmd.req.link_mode_masks_nwords);

    return send_ioctl(ctx, &ecmd);
}

static struct ethtool_link_usettings *
do_ioctl_gset(struct cmd_context *ctx)
{
    int err;

```

```

struct ethtool_cmd ecmd;
struct ethtool_link_usettings *link_usettings;

memset(&ecmd, 0, sizeof(ecmd));
ecmd.cmd = ETHTOOL_GSET;
err = send_ioctl(ctx, &ecmd);
if (err < 0)
return NULL;

link_usettings = calloc(1, sizeof(*link_usettings));
if (link_usettings == NULL)
return NULL;

/* remember that ETHTOOL_GSET was used */
link_usettings->base.cmd = ETHTOOL_GSET;

link_usettings->base.link_mode_masks_nwords = 1;
link_usettings->link_modes.supported[0] = ecmd.supported;
link_usettings->link_modes.advertising[0] = ecmd.advertising;
link_usettings->link_modes.lp_advertising[0] = ecmd.lp_advertising;
link_usettings->base.speed = ethtool_cmd_speed(&ecmd);
link_usettings->base.duplex = ecmd.duplex;
link_usettings->base.port = ecmd.port;
link_usettings->base.phy_address = ecmd.phy_address;
link_usettings->deprecated.transceiver = ecmd.transceiver;
link_usettings->base.autoneg = ecmd.autoneg;
link_usettings->base.mdio_support = ecmd.mdio_support;
/* ignored (fully deprecated): maxrxpkt, maxtxpkt */
link_usettings->base.eth_tp_mdix = ecmd.eth_tp_mdix;
link_usettings->base.eth_tp_mdix_ctrl = ecmd.eth_tp_mdix_ctrl;

return link_usettings;
}

static bool ethtool_link_mode_is_backward_compatible(const u32 *mask)
{
    unsigned int i;

    for (i = 1; i < ETHTOOL_LINK_MODE_MASK_MAX_KERNEL_NU32; ++i)
        if (mask[i])
            return false;

    return true;
}

static int
do_ioctl_sset(struct cmd_context *ctx,
               const struct ethtool_link_usettings *link_usettings)
{
    struct ethtool_cmd ecmd;

    /* refuse to send ETHTOOL_SSET ioctl if link_usettings was
     * retrieved with ETHTOOL_GLINKSETTINGS
     */

```

```

if (link_usettings->base.cmd != ETHTOOL_GSET)
return -1;

if (link_usettings->base.link_mode_masks_nwords <= 0)
return -1;

/* refuse to sset if any bit > 31 is set */
if (!ethtool_link_mode_is_backward_compatible(
    link_usettings->link_modes.supported))
return -1;
if (!ethtool_link_mode_is_backward_compatible(
    link_usettings->link_modes.advertising))
return -1;
if (!ethtool_link_mode_is_backward_compatible(
    link_usettings->link_modes.lp_advertising))
return -1;

memset(&ecmd, 0, sizeof(ecmd));
ecmd.cmd = ETHTOOL_SSET;

ecmd.supported = link_usettings->link_modes.supported[0];
ecmd.advertising = link_usettings->link_modes.advertising[0];
ecmd.lp_advertising = link_usettings->link_modes.lp_advertising[0];
ethtool_cmd_speed_set(&ecmd, link_usettings->base.speed);
ecmd.duplex = link_usettings->base.duplex;
ecmd.port = link_usettings->base.port;
ecmd.phy_address = link_usettings->base.phy_address;
ecmd.transceiver = link_usettings->deprecated.transceiver;
ecmd.autoneg = link_usettings->base.autoneg;
ecmd.mdio_support = link_usettings->base.mdio_support;
/* ignored (fully deprecated): maxrxpkt, maxtxpkt */
ecmd.eth_tp_mdix = link_usettings->base.eth_tp_mdix;
ecmd.eth_tp_mdix_ctrl = link_usettings->base.eth_tp_mdix_ctrl;
return send_ioctl(ctx, &ecmd);
}

static int do_gset(struct cmd_context *ctx)
{
    int err;
    struct ethtool_link_usettings *link_usettings;
    struct ethtool_wolinfo wolinfo;
    struct ethtool_value edata;
    int allfail = 1;

    if (ctx->argc != 0)
        exit_bad_args();

    fprintf(stdout, "Settings for %s:\n", ctx->devname);

    link_usettings = do_ioctl_glinksettings(ctx);
    if (link_usettings == NULL)
        link_usettings = do_ioctl_gset(ctx);
    if (link_usettings != NULL) {
        err = dump_link_usettings(link_usettings);
        free(link_usettings);
    }

```

```
if (err)
return err;
allfail = 0;
} else if (errno != EOPNOTSUPP) {
perror("Cannot get device settings");
}

wolinfo.cmd = ETHTOOL_GWOL;
err = send_ioctl(ctx, &wolinfo);
if (err == 0) {
err = dump_wol(&wolinfo);
if (err)
return err;
allfail = 0;
} else if (errno != EOPNOTSUPP) {
perror("Cannot get wake-on-lan settings");
}

edata.cmd = ETHTOOL_GMSGLVL;
err = send_ioctl(ctx, &edata);
if (err == 0) {
fprintf(stdout, "Current message level: 0x%08x (%d)\n",
        " ",
        edata.data, edata.data);
print_flags(flags_msglvl, ARRAY_SIZE(flags_msglvl),
        edata.data);
fprintf(stdout, "\n");
allfail = 0;
} else if (errno != EOPNOTSUPP) {
perror("Cannot get message level");
}

edata.cmd = ETHTOOL_GLINK;
err = send_ioctl(ctx, &edata);
if (err == 0) {
fprintf(stdout, "Link detected: %s\n",
        edata.data ? "yes": "no");
allfail = 0;
} else if (errno != EOPNOTSUPP) {
perror("Cannot get link status");
}

if (allfail) {
fprintf(stdout, "No data available\n");
return 75;
}
return 0;
}

static int do_sset(struct cmd_context *ctx)
{
int speed_wanted = -1;
int duplex_wanted = -1;
int port_wanted = -1;
```

```

int mdix_wanted = -1;
int autoneg_wanted = -1;
int phyad_wanted = -1;
int xcvr_wanted = -1;
u32 *full_advertising_wanted = NULL;
u32 *advertising_wanted = NULL;
ETHTOOL_DECLARE_LINK_MODE_MASK(mask_full_advertising_wanted);
ETHTOOL_DECLARE_LINK_MODE_MASK(mask_advertising_wanted);
int gset_changed = 0; /* did anything in GSET change? */
u32 wol_wanted = 0;
int wol_change = 0;
u8 sopass_wanted[SOPASS_MAX];
int sopass_change = 0;
int gwol_changed = 0; /* did anything in GWOL change? */
int msglvl_changed = 0;
u32 msglvl_wanted = 0;
u32 msglvl_mask = 0;
struct cmdline_info cmdline_msglvl[ARRAY_SIZE(flags_msglvl)];
int argc = ctx->argc;
char **argp = ctx->argp;
int i;
int err = 0;

for (i = 0; i < ARRAY_SIZE(flags_msglvl); i++)
flag_to_cmdline_info(flags_msglvl[i].name,
    flags_msglvl[i].value,
    &msglvl_wanted, &msglvl_mask,
    &cmdline_msglvl[i]);

for (i = 0; i < argc; i++) {
if (!strcmp(argp[i], "speed")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
speed_wanted = get_int(argp[i], 10);
} else if (!strcmp(argp[i], "duplex")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
if (!strcmp(argp[i], "half"))
duplex_wanted = DUPLEX_HALF;
else if (!strcmp(argp[i], "full"))
duplex_wanted = DUPLEX_FULL;
else
exit_bad_args();
} else if (!strcmp(argp[i], "port")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
if (!strcmp(argp[i], "tp"))
port_wanted = PORT_TP;
else if (!strcmp(argp[i], "aui"))

```

```

port_wanted = PORT_AUI;
else if (!strcmp(argp[i], "bnc"))
port_wanted = PORT_BNC;
else if (!strcmp(argp[i], "mii"))
port_wanted = PORT_MII;
else if (!strcmp(argp[i], "fibre"))
port_wanted = PORT_FIBRE;
else
exit_bad_args();
} else if (!strcmp(argp[i], "mdix")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
if (!strcmp(argp[i], "auto"))
mdix_wanted = ETH_TP_MDI_AUTO;
else if (!strcmp(argp[i], "on"))
mdix_wanted = ETH_TP_MDI_X;
else if (!strcmp(argp[i], "off"))
mdix_wanted = ETH_TP_MDI;
else
exit_bad_args();
} else if (!strcmp(argp[i], "autoneg")) {
i += 1;
if (i >= argc)
exit_bad_args();
if (!strcmp(argp[i], "on")) {
gset_changed = 1;
autoneg_wanted = AUTONEG_ENABLE;
} else if (!strcmp(argp[i], "off")) {
gset_changed = 1;
autoneg_wanted = AUTONEG_DISABLE;
} else {
exit_bad_args();
}
} else if (!strcmp(argp[i], "advertise")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
if (parse_hex_u32_bitmap(
    argp[i],
    ETHTOOL_LINK_MODE_MASK_MAX_KERNEL_NBITS,
    mask_full_advertising_wanted))
exit_bad_args();
full_advertising_wanted = mask_full_advertising_wanted;
} else if (!strcmp(argp[i], "phyad")) {
gset_changed = 1;
i += 1;
if (i >= argc)
exit_bad_args();
phyad_wanted = get_int(argp[i], 0);
} else if (!strcmp(argp[i], "xcvr")) {
gset_changed = 1;

```

```

i += 1;
if (i >= argc)
exit_bad_args();
if (!strcmp(argp[i], "internal"))
xcvr_wanted = XCVR_INTERNAL;
else if (!strcmp(argp[i], "external"))
xcvr_wanted = XCVR_EXTERNAL;
else
exit_bad_args();
} else if (!strcmp(argp[i], "wol")) {
gwo_l_changed = 1;
i++;
if (i >= argc)
exit_bad_args();
if (parse_wolopts(argp[i], &wol_wanted) < 0)
exit_bad_args();
wol_change = 1;
} else if (!strcmp(argp[i], "sopass")) {
gwo_l_changed = 1;
i++;
if (i >= argc)
exit_bad_args();
get_mac_addr(argp[i], sopass_wanted);
sopass_change = 1;
} else if (!strcmp(argp[i], "msglvl")) {
i++;
if (i >= argc)
exit_bad_args();
if (isdigit((unsigned char)argp[i][0])) {
msglvl_changed = 1;
msglvl_mask = ~0;
msglvl_wanted =
get_uint_range(argp[i], 0,
0xffffffff);
} else {
ctx->argc -= i;
ctx->argp += i;
parse_generic_cmdline(
ctx, &msglvl_changed,
cmdline_msglvl,
ARRAY_SIZE(cmdline_msglvl));
break;
}
} else {
exit_bad_args();
}
}

if (full_advertising_wanted == NULL) {
/* User didn't supply a full advertisement bitfield:
 * construct one from the specified speed and duplex.
 */
int adv_bit = -1;

if (speed_wanted == SPEED_10 && duplex_wanted == DUPLEX_HALF)

```



```

adv_bit = ETHTOOL_LINK_MODE_10baseT_Half_BIT;
else if (speed_wanted == SPEED_10 &&
duplex_wanted == DUPLEX_FULL)
adv_bit = ETHTOOL_LINK_MODE_10baseT_Full_BIT;
else if (speed_wanted == SPEED_100 &&
duplex_wanted == DUPLEX_HALF)
adv_bit = ETHTOOL_LINK_MODE_100baseT_Half_BIT;
else if (speed_wanted == SPEED_100 &&
duplex_wanted == DUPLEX_FULL)
adv_bit = ETHTOOL_LINK_MODE_100baseT_Full_BIT;
else if (speed_wanted == SPEED_1000 &&
duplex_wanted == DUPLEX_HALF)
adv_bit = ETHTOOL_LINK_MODE_1000baseT_Half_BIT;
else if (speed_wanted == SPEED_1000 &&
duplex_wanted == DUPLEX_FULL)
adv_bit = ETHTOOL_LINK_MODE_1000baseT_Full_BIT;
else if (speed_wanted == SPEED_2500 &&
duplex_wanted == DUPLEX_FULL)
adv_bit = ETHTOOL_LINK_MODE_2500baseX_Full_BIT;
else if (speed_wanted == SPEED_10000 &&
duplex_wanted == DUPLEX_FULL)
adv_bit = ETHTOOL_LINK_MODE_10000baseT_Full_BIT;

if (adv_bit >= 0) {
    advertising_wanted = mask_advertising_wanted;
    ethtool_link_mode_zero(advertising_wanted);
    ethtool_link_mode_set_bit(
        adv_bit, advertising_wanted);
}
/* otherwise: auto negotiate without forcing,
 * all supported speed will be assigned below
 */
}

if (gset_changed) {
    struct ethtool_link_usettings *link_usettings;

    link_usettings = do_ioctl_glinksettings(ctx);
    if (link_usettings == NULL)
        link_usettings = do_ioctl_gset(ctx);
    if (link_usettings == NULL) {
        perror("Cannot get current device settings");
        err = -1;
    } else {
        /* Change everything the user specified. */
        if (speed_wanted != -1)
            link_usettings->base.speed = speed_wanted;
        if (duplex_wanted != -1)
            link_usettings->base.duplex = duplex_wanted;
        if (port_wanted != -1)
            link_usettings->base.port = port_wanted;
        if (mdix_wanted != -1) {
            /* check driver supports MDI-X */
            if (link_usettings->base.eth_tp_mdix_ctrl

```

```

    != ETH_TP_MDI_INVALID)
link_usettings->base.eth_tp_mdix_ctrl
= mdix_wanted;
else
fprintf(stderr,
"setting MDI not supported\n");
}
if (autoneg_wanted != -1)
link_usettings->base.autoneg = autoneg_wanted;
if (phyad_wanted != -1)
link_usettings->base.phy_address = phyad_wanted;
if (xcvr_wanted != -1)
link_usettings->deprecated.transceiver
= xcvr_wanted;
/* XXX If the user specified speed or duplex
 * then we should mask the advertised modes
 * accordingly. For now, warn that we aren't
 * doing that.
 */
if ((speed_wanted != -1 || duplex_wanted != -1) &&
    link_usettings->base.autoneg &&
    advertising_wanted == NULL) {
fprintf(stderr, "Cannot advertise");
if (speed_wanted >= 0)
fprintf(stderr, " speed %d",
speed_wanted);
if (duplex_wanted >= 0)
fprintf(stderr, " duplex %s",
duplex_wanted ?
"full" : "half");
fprintf(stderr, "\n");
}
if (autoneg_wanted == AUTONEG_ENABLE &&
    advertising_wanted == NULL &&
    full_advertising_wanted == NULL) {
unsigned int i;

/* Auto negotiation enabled, but with
 * unspecified speed and duplex: enable all
 * supported speeds and duplexes.
 */
ethtool_link_mode_for_each_u32(i) {
u32 sup = link_usettings->link_modes.supported[i];
u32 *adv = link_usettings->link_modes.advertising + i;

*adv = ((*adv & ~all_advertised_modes[i])
| (sup & all_advertised_modes[i]));
}

/* If driver supports unknown flags, we cannot
 * be sure that we enable all link modes.
 */
ethtool_link_mode_for_each_u32(i) {
u32 sup = link_usettings->link_modes.supported[i];

```

```

if ((sup & all_advertised_flags[i]) != sup) {
    fprintf(stderr, "Driver supports one or more unknown flags\n");
    break;
}
} else if (advertising_wanted != NULL) {
    unsigned int i;

    /* Enable all requested modes */
    ethtool_link_mode_for_each_u32(i) {
        u32 *adv = link_usettings->link_modes.advertising + i;

        *adv = ((*adv & ~all_advertised_modes[i])
| advertising_wanted[i]);
    }
    } else if (full_advertising_wanted != NULL) {
        ethtool_link_mode_copy(
            link_usettings->link_modes.advertising,
            full_advertising_wanted);
    }

    /* Try to perform the update. */
    if (link_usettings->base.cmd == ETHTOOL_GLINKSETTINGS)
        err = do_ioctl_slinksettings(ctx,
            link_usettings);
    else
        err = do_ioctl_sset(ctx, link_usettings);
    free(link_usettings);
    if (err < 0)
        perror("Cannot set new settings");
    }
    if (err < 0) {
        if (speed_wanted != -1)
            fprintf(stderr, " not setting speed\n");
        if (duplex_wanted != -1)
            fprintf(stderr, " not setting duplex\n");
        if (port_wanted != -1)
            fprintf(stderr, " not setting port\n");
        if (autoneg_wanted != -1)
            fprintf(stderr, " not setting autoneg\n");
        if (phyad_wanted != -1)
            fprintf(stderr, " not setting phy_address\n");
        if (xcvr_wanted != -1)
            fprintf(stderr, " not setting transceiver\n");
        if (mdix_wanted != -1)
            fprintf(stderr, " not setting mdix\n");
    }
    }

    if (gwol_changed) {
        struct ethtool_wolinfo wol;

        wol.cmd = ETHTOOL_GWOL;
        err = send_ioctl(ctx, &wol);
    }

```

```

if (err < 0) {
perror("Cannot get current wake-on-lan settings");
} else {
/* Change everything the user specified. */
if (wol_change) {
wol.wolopts = wol_wanted;
}
if (sopass_change) {
int i;
for (i = 0; i < SOPASS_MAX; i++) {
wol.sopass[i] = sopass_wanted[i];
}
}

/* Try to perform the update. */
wol.cmd = ETHTOOL_SWOL;
err = send_ioctl(ctx, &wol);
if (err < 0)
perror("Cannot set new wake-on-lan settings");
}
if (err < 0) {
if (wol_change)
fprintf(stderr, " not setting wol\n");
if (sopass_change)
fprintf(stderr, " not setting sopass\n");
}
}

if (msglvl_changed) {
struct ethtool_value edata;

edata.cmd = ETHTOOL_GMSGLVL;
err = send_ioctl(ctx, &edata);
if (err < 0) {
perror("Cannot get msglvl");
} else {
edata.cmd = ETHTOOL_SMSGLVL;
edata.data = ((edata.data & ~msglvl_mask) |
msglvl_wanted);
err = send_ioctl(ctx, &edata);
if (err < 0)
perror("Cannot set new msglvl");
}
}

return 0;
}

static int do_gregs(struct cmd_context *ctx)
{
int gregs_changed = 0;
int gregs_dump_raw = 0;
int gregs_dump_hex = 0;
char *gregs_dump_file = NULL;
struct cmdline_info cmdline_gregs[] = {

```

```

{ "raw", CMDL_BOOL, &gregs_dump_raw, NULL },
{ "hex", CMDL_BOOL, &gregs_dump_hex, NULL },
{ "file", CMDL_STR, &gregs_dump_file, NULL },
};
int err;
struct ethtool_drvinfo drvinfo;
struct ethtool_regs *regs;

parse_generic_cmdline(ctx, &gregs_changed,
    cmdline_gregs, ARRAY_SIZE(cmdline_gregs));

drvinfo.cmd = ETHTOOL_GDRVINFO;
err = send_ioctl(ctx, &drvinfo);
if (err < 0) {
    perror("Cannot get driver information");
    return 72;
}

regs = calloc(1, sizeof(*regs)+drvinfo.regdump_len);
if (!regs) {
    perror("Cannot allocate memory for register dump");
    return 73;
}
regs->cmd = ETHTOOL_GREGS;
regs->len = drvinfo.regdump_len;
err = send_ioctl(ctx, regs);
if (err < 0) {
    perror("Cannot get register dump");
    free(regs);
    return 74;
}

if (!gregs_dump_raw && gregs_dump_file != NULL) {
    /* overwrite reg values from file dump */
    FILE *f = fopen(gregs_dump_file, "r");
    struct stat st;
    size_t nread;

    if (!f || fstat(fileno(f), &st) < 0) {
        fprintf(stderr, "Can't open '%s': %s\n",
            gregs_dump_file, strerror(errno));
        free(regs);
        return 75;
    }

    regs = realloc(regs, sizeof(*regs) + st.st_size);
    regs->len = st.st_size;
    nread = fread(regs->data, regs->len, 1, f);
    fclose(f);
    if (nread != 1) {
        free(regs);
        return 75;
    }
}

```

```

if (dump_regs(gregs_dump_raw, gregs_dump_hex,
    &drvinfo, regs) < 0) {
    fprintf(stderr, "Cannot dump registers\n");
    free(regs);
    return 75;
}
free(regs);

return 0;
}

static int do_nway_rst(struct cmd_context *ctx)
{
    struct ethtool_value edata;
    int err;

    if (ctx->argc != 0)
        exit_bad_args();

    edata.cmd = ETHTOOL_NWAY_RST;
    err = send_ioctl(ctx, &edata);
    if (err < 0)
        perror("Cannot restart autonegotiation");

    return err;
}

static int do_geeprom(struct cmd_context *ctx)
{
    {
        int geeprom_changed = 0;
        int geeprom_dump_raw = 0;
        u32 geeprom_offset = 0;
        u32 geeprom_length = -1;
        struct cmdline_info cmdline_geeprom[] = {
            { "offset", CMDL_U32, &geeprom_offset, NULL },
            { "length", CMDL_U32, &geeprom_length, NULL },
            { "raw", CMDL_BOOL, &geeprom_dump_raw, NULL },
        };
        int err;
        struct ethtool_drvinfo drvinfo;
        struct ethtool_eeprom *eeprom;

        parse_generic_cmdline(ctx, &geeprom_changed,
            cmdline_geeprom, ARRAY_SIZE(cmdline_geeprom));

        drvinfo.cmd = ETHTOOL_GDRVINFO;
        err = send_ioctl(ctx, &drvinfo);
        if (err < 0) {
            perror("Cannot get driver information");
            return 74;
        }

        if (geeprom_length == -1)
            geeprom_length = drvinfo.eedump_len;
    }
}

```

```

if (drvinfo.eedump_len < geeprom_offset + geeprom_length)
    geeprom_length = drvinfo.eedump_len - geeprom_offset;

eeprom = calloc(1, sizeof(*eeprom)+geeprom_length);
if (!eeprom) {
    perror("Cannot allocate memory for EEPROM data");
    return 75;
}
eeprom->cmd = ETHTOOL_GEEPROM;
eeprom->len = geeprom_length;
eeprom->offset = geeprom_offset;
err = send_ioctl(ctx, eeprom);
if (err < 0) {
    perror("Cannot get EEPROM data");
    free(eeprom);
    return 74;
}
err = dump_eeprom(geeprom_dump_raw, &drvinfo, eeprom);
free(eeprom);

return err;
}

static int do_seeprom(struct cmd_context *ctx)
{
    int seeprom_changed = 0;
    u32 seeprom_magic = 0;
    u32 seeprom_length = -1;
    u32 seeprom_offset = 0;
    u8 seeprom_value = 0;
    int seeprom_value_seen = 0;
    struct cmdline_info cmdline_seeprom[] = {
        { "magic", CMDL_U32, &seeprom_magic, NULL },
        { "offset", CMDL_U32, &seeprom_offset, NULL },
        { "length", CMDL_U32, &seeprom_length, NULL },
        { "value", CMDL_U8, &seeprom_value, NULL,
          0, &seeprom_value_seen },
    };
    int err;
    struct ethtool_drvinfo drvinfo;
    struct ethtool_eeprom *eeprom;

    parse_generic_cmdline(ctx, &seeprom_changed,
        cmdline_seeprom, ARRAY_SIZE(cmdline_seeprom));

    drvinfo.cmd = ETHTOOL_GDRVINFO;
    err = send_ioctl(ctx, &drvinfo);
    if (err < 0) {
        perror("Cannot get driver information");
        return 74;
    }

    if (seeprom_value_seen)

```

```

seeprom_length = 1;

if (seeprom_length == -1)
seeprom_length = drvinfo.eedump_len;

if (drvinfo.eedump_len < seeprom_offset + seeprom_length) {
fprintf(stderr, "offset & length out of bounds\n");
return 1;
}

eeprom = calloc(1, sizeof(*eeprom)+seeprom_length);
if (!eeprom) {
perror("Cannot allocate memory for EEPROM data");
return 75;
}

eeprom->cmd = ETHTOOL_SEEPROM;
eeprom->len = seeprom_length;
eeprom->offset = seeprom_offset;
eeprom->magic = seeprom_magic;
eeprom->data[0] = seeprom_value;

/* Multi-byte write: read input from stdin */
if (!seeprom_value_seen) {
if (fread(eeprom->data, eeprom->len, 1, stdin) != 1) {
fprintf(stderr, "not enough data from stdin\n");
free(eeprom);
return 75;
}
if ((fgetc(stdin) != EOF) || !feof(stdin)) {
fprintf(stderr, "too much data from stdin\n");
free(eeprom);
return 75;
}
}

err = send_ioctl(ctx, eeprom);
if (err < 0) {
perror("Cannot set EEPROM data");
err = 87;
}
free(eeprom);

return err;
}

static int do_test(struct cmd_context *ctx)
{
enum {
ONLINE=0,
OFFLINE,
EXTERNAL_LB,
} test_type;
int err;
struct ethtool_test *test;

```



```
struct ethtool_gstrings *strings;

if (ctx->argc > 1)
    exit_bad_args();
if (ctx->argc == 1) {
    if (!strcmp(ctx->argv[0], "online")) {
        test_type = ONLINE;
    } else if (!strcmp(*ctx->argv, "offline")) {
        test_type = OFFLINE;
    } else if (!strcmp(*ctx->argv, "external_lb")) {
        test_type = EXTERNAL_LB;
    } else {
        exit_bad_args();
    }
} else {
    test_type = OFFLINE;
}

strings = get_stringset(ctx, ETH_SS_TEST,
    offsetof(struct ethtool_drvinfo, testinfo_len),
    1);
if (!strings) {
    perror("Cannot get strings");
    return 74;
}

test = calloc(1, sizeof(*test) + strings->len * sizeof(u64));
if (!test) {
    perror("Cannot allocate memory for test info");
    free(strings);
    return 73;
}
memset(test->data, 0, strings->len * sizeof(u64));
test->cmd = ETHTOOL_TEST;
test->len = strings->len;
if (test_type == EXTERNAL_LB)
    test->flags = (ETH_TEST_FL_OFFLINE | ETH_TEST_FL_EXTERNAL_LB);
else if (test_type == OFFLINE)
    test->flags = ETH_TEST_FL_OFFLINE;
else
    test->flags = 0;
err = send_ioctl(ctx, test);
if (err < 0) {
    perror("Cannot test");
    free(test);
    free(strings);
    return 74;
}

err = dump_test(test, strings);
free(test);
free(strings);

return err;
```

```

    }

    static int do_phys_id(struct cmd_context *ctx)
    {
        int err;
        struct ethtool_value edata;
        int phys_id_time;

        if (ctx->argc > 1)
            exit_bad_args();
        if (ctx->argc == 1)
            phys_id_time = get_int(*ctx->argv, 0);
        else
            phys_id_time = 0;

        edata.cmd = ETHTOOL_PHYS_ID;
        edata.data = phys_id_time;
        err = send_ioctl(ctx, &edata);
        if (err < 0)
            perror("Cannot identify NIC");

        return err;
    }

    static int do_gstats(struct cmd_context *ctx, int cmd, int stringset,
        const char *name)
    {
        struct ethtool_gstrings *strings;
        struct ethtool_stats *stats;
        unsigned int n_stats, sz_stats, i;
        int err;

        if (ctx->argc != 0)
            exit_bad_args();

        strings = get_stringset(ctx, stringset,
            offsetof(struct ethtool_drvinfo, n_stats),
            0);
        if (!strings) {
            perror("Cannot get stats strings information");
            return 96;
        }

        n_stats = strings->len;
        if (n_stats < 1) {
            fprintf(stderr, "no stats available\n");
            free(strings);
            return 94;
        }

        sz_stats = n_stats * sizeof(u64);

        stats = calloc(1, sz_stats + sizeof(struct ethtool_stats));
        if (!stats) {
            fprintf(stderr, "no memory available\n");

```

```

    free(strings);
    return 95;
}

stats->cmd = cmd;
stats->n_stats = n_stats;
err = send_ioctl(ctx, stats);
if (err < 0) {
    perror("Cannot get stats information");
    free(strings);
    free(stats);
    return 97;
}

/* todo - pretty-print the strings per-driver */
fprintf(stdout, "%s statistics:\n", name);
for (i = 0; i < n_stats; i++) {
    fprintf(stdout, "    %s: %llu\n",
        ETH_GSTRING_LEN,
        &strings->data[i * ETH_GSTRING_LEN],
        stats->data[i]);
}
free(strings);
free(stats);

return 0;
}

static int do_gnicstats(struct cmd_context *ctx)
{
    return do_gstats(ctx, ETHTOOL_GSTATS, ETH_SS_STATS, "NIC");
}

static int do_gphystats(struct cmd_context *ctx)
{
    return do_gstats(ctx, ETHTOOL_GPHYSTATS, ETH_SS_PHY_STATS, "PHY");
}

static int do_srxntuple(struct cmd_context *ctx,
    struct ethtool_rx_flow_spec *rx_rule_fs);

static int do_srxclass(struct cmd_context *ctx)
{
    int err;

    if (ctx->argc < 2)
        exit_bad_args();

    if (ctx->argc == 3 && !strcmp(ctx->argp[0], "rx-flow-hash")) {
        int rx_fhash_set;
        u32 rx_fhash_val;
        struct ethtool_rxnfc nfccmd;

        rx_fhash_set = rxflow_str_to_type(ctx->argp[1]);
    }
}

```

```

if (!rx_fhash_set)
exit_bad_args();
if (parse_rxhashopts(ctx->argp[2], &rx_fhash_val) < 0)
exit_bad_args();

nfccmd.cmd = ETHTOOL_SRXFH;
nfccmd.flow_type = rx_fhash_set;
nfccmd.data = rx_fhash_val;

err = send_ioctl(ctx, &nfccmd);
if (err < 0)
perror("Cannot change RX network flow hashing options");
} else if (!strcmp(ctx->argp[0], "flow-type")) {
struct ethtool_rx_flow_spec rx_rule_fs;

ctx->argc--;
ctx->argp++;
if (rxclass_parse_ruleopts(ctx, &rx_rule_fs) < 0)
exit_bad_args();

/* attempt to add rule via N-tuple specifier */
err = do_srxntuple(ctx, &rx_rule_fs);
if (!err)
return 0;

/* attempt to add rule via network flow classifier */
err = rxclass_rule_ins(ctx, &rx_rule_fs);
if (err < 0) {
fprintf(stderr, "Cannot insert"
" classification rule\n");
return 1;
}
} else if (!strcmp(ctx->argp[0], "delete")) {
int rx_class_rule_del =
get_uint_range(ctx->argp[1], 0, INT_MAX);

err = rxclass_rule_del(ctx, rx_class_rule_del);

if (err < 0) {
fprintf(stderr, "Cannot delete"
" classification rule\n");
return 1;
}
} else {
exit_bad_args();
}

return 0;
}

static int do_grxclass(struct cmd_context *ctx)
{
struct ethtool_rxnfc nfccmd;
int err;

```

```

if (ctx->argc == 2 && !strcmp(ctx->argp[0], "rx-flow-hash")) {
    int rx_fhash_get;

    rx_fhash_get = rxflow_str_to_type(ctx->argp[1]);
    if (!rx_fhash_get)
        exit_bad_args();

    nfccmd.cmd = ETHTOOL_GRXFB;
    nfccmd.flow_type = rx_fhash_get;
    err = send_ioctl(ctx, &nfccmd);
    if (err < 0)
        perror("Cannot get RX network flow hashing options");
    else
        dump_rxhash(rx_fhash_get, nfccmd.data);
} else if (ctx->argc == 2 && !strcmp(ctx->argp[0], "rule")) {
    int rx_class_rule_get =
        get_uint_range(ctx->argp[1], 0, INT_MAX);

    err = rxclass_rule_get(ctx, rx_class_rule_get);
    if (err < 0)
        fprintf(stderr, "Cannot get RX classification rule\n");
    } else if (ctx->argc == 0) {
        nfccmd.cmd = ETHTOOL_GRXRINGS;
        err = send_ioctl(ctx, &nfccmd);
        if (err < 0)
            perror("Cannot get RX rings");
        else
            fprintf(stdout, "%d RX rings available\n",
                (int)nfccmd.data);

        err = rxclass_rule_getall(ctx);
        if (err < 0)
            fprintf(stderr, "RX classification rule retrieval failed\n");

    } else {
        exit_bad_args();
    }

    return err ? 1 : 0;
}

static void print_indir_table(struct cmd_context *ctx,
    struct ethtool_rxnfc *ring_count,
    u32 indir_size, u32 *indir)
{
    u32 i;

    printf("RX flow hash indirection table for %s with %llu RX ring(s):\n",
        ctx->devname, ring_count->data);

    if (!indir_size)
        printf("Operation not supported\n");

    for (i = 0; i < indir_size; i++) {

```

```

    if (i % 8 == 0)
        printf("%5u: ", i);
    printf(" %5u", indir[i]);
    if (i % 8 == 7 || i == indir_size - 1)
        fputc('\n', stdout);
    }
}

static int do_grxfhindir(struct cmd_context *ctx,
    struct ethtool_rxnfc *ring_count)
{
    struct ethtool_rxfh_indir indir_head;
    struct ethtool_rxfh_indir *indir;
    int err;

    indir_head.cmd = ETHTOOL_GRXFHINDIR;
    indir_head.size = 0;
    err = send_ioctl(ctx, &indir_head);
    if (err < 0) {
        perror("Cannot get RX flow hash indirection table size");
        return 1;
    }

    indir = malloc(sizeof(*indir) +
        indir_head.size * sizeof(*indir->ring_index));
    if (!indir) {
        perror("Cannot allocate memory for indirection table");
        return 1;
    }

    indir->cmd = ETHTOOL_GRXFHINDIR;
    indir->size = indir_head.size;
    err = send_ioctl(ctx, indir);
    if (err < 0) {
        perror("Cannot get RX flow hash indirection table");
        free(indir);
        return 1;
    }

    print_indir_table(ctx, ring_count, indir->size, indir->ring_index);

    free(indir);
    return 0;
}

static int do_grxfh(struct cmd_context *ctx)
{
    struct ethtool_rxfh rss_head = {0};
    struct ethtool_rxnfc ring_count;
    struct ethtool_rxfh *rss;
    u32 i, indir_bytes;
    char *hkey;
    int err;

    ring_count.cmd = ETHTOOL_GRXRINGS;

```

```

err = send_ioctl(ctx, &ring_count);
if (err < 0) {
perror("Cannot get RX ring count");
return 1;
}

rss_head.cmd = ETHTOOL_GRSSH;
err = send_ioctl(ctx, &rss_head);
if (err < 0 && errno == EOPNOTSUPP) {
return do_grxfhindir(ctx, &ring_count);
} else if (err < 0) {
perror("Cannot get RX flow hash indir size and/or key size");
return 1;
}

rss = calloc(1, sizeof(*rss) +
rss_head.indir_size * sizeof(rss_head.rss_config[0]) +
rss_head.key_size);
if (!rss) {
perror("Cannot allocate memory for RX flow hash config");
return 1;
}

rss->cmd = ETHTOOL_GRSSH;
rss->indir_size = rss_head.indir_size;
rss->key_size = rss_head.key_size;
err = send_ioctl(ctx, rss);
if (err < 0) {
perror("Cannot get RX flow hash configuration");
free(rss);
return 1;
}

print_indir_table(ctx, &ring_count, rss->indir_size, rss->rss_config);

indir_bytes = rss->indir_size * sizeof(rss->rss_config[0]);
hkey = ((char *)rss->rss_config + indir_bytes);

printf("RSS hash key:\n");
if (!rss->key_size)
printf("Operation not supported\n");

for (i = 0; i < rss->key_size; i++) {
if (i == (rss->key_size - 1))
printf("%02x\n", (u8) hkey[i]);
else
printf("%02x:", (u8) hkey[i]);
}

free(rss);
return 0;
}

static int fill_indir_table(u32 *indir_size, u32 *indir, int rxfhindir_default,

```

```

    int rxfhindir_equal, char **rxfhindir_weight,
    u32 num_weights)
{
    u32 i;
    if (rxfhindir_equal) {
        for (i = 0; i < *indir_size; i++)
            indir[i] = i % rxfhindir_equal;
    } else if (rxfhindir_weight) {
        u32 j, weight, sum = 0, partial = 0;

        for (j = 0; j < num_weights; j++) {
            weight = get_u32(rxfhindir_weight[j], 0);
            sum += weight;
        }

        if (sum == 0) {
            fprintf(stderr,
                "At least one weight must be non-zero\n");
            return 2;
        }

        if (sum > *indir_size) {
            fprintf(stderr,
                "Total weight exceeds the size of the "
                "indirection table\n");
            return 2;
        }

        j = -1;
        for (i = 0; i < *indir_size; i++) {
            while (i >= (*indir_size) * partial / sum) {
                j += 1;
                weight = get_u32(rxfhindir_weight[j], 0);
                partial += weight;
            }
            indir[i] = j;
        }
    } else if (rxfhindir_default) {
        /* "*indir_size == 0" ==> reset indir to default */
        *indir_size = 0;
    } else {
        *indir_size = ETH_RXFH_INDIR_NO_CHANGE;
    }

    return 0;
}

static int do_srxfhindir(struct cmd_context *ctx, int rxfhindir_default,
    int rxfhindir_equal, char **rxfhindir_weight,
    u32 num_weights)
{
    struct ethtool_rxfh_indir indir_head;
    struct ethtool_rxfh_indir *indir;
    int err;

```



```
indir_head.cmd = ETHTOOL_GRXFHINDIR;
indir_head.size = 0;
err = send_ioctl(ctx, &indir_head);
if (err < 0) {
    perror("Cannot get RX flow hash indirection table size");
    return 1;
}

indir = malloc(sizeof(*indir) +
               indir_head.size * sizeof(*indir->ring_index));

if (!indir) {
    perror("Cannot allocate memory for indirection table");
    return 1;
}

indir->cmd = ETHTOOL_SRXFHINDIR;
indir->size = indir_head.size;

if (fill_indir_table(&indir->size, indir->ring_index,
                    rxfhindir_default, rxfhindir_equal,
                    rxfhindir_weight, num_weights)) {
    free(indir);
    return 1;
}

err = send_ioctl(ctx, indir);
if (err < 0) {
    perror("Cannot set RX flow hash indirection table");
    free(indir);
    return 1;
}

free(indir);
return 0;
}

static int do_srxfh(struct cmd_context *ctx)
{
    struct ethtool_rxfh rss_head = {0};
    struct ethtool_rxfh *rss;
    struct ethtool_rxnfc ring_count;
    int rxfhindir_equal = 0, rxfhindir_default = 0;
    char **rxfhindir_weight = NULL;
    char *rxfhindir_key = NULL;
    char *hkey = NULL;
    int err = 0;
    u32 arg_num = 0, indir_bytes = 0;
    u32 entry_size = sizeof(rss_head.rss_config[0]);
    u32 num_weights = 0;

    if (ctx->argc < 1)
        exit_bad_args();
}
```

```

while (arg_num < ctx->argc) {
if (!strcmp(ctx->argv[arg_num], "equal")) {
++arg_num;
rxfhindir_equal = get_int_range(ctx->argv[arg_num],
0, 1, INT_MAX);
++arg_num;
} else if (!strcmp(ctx->argv[arg_num], "weight")) {
++arg_num;
rxfhindir_weight = ctx->argv + arg_num;
while (arg_num < ctx->argc &&
isdigit((unsigned char)ctx->argv[arg_num][0])) {
++arg_num;
++num_weights;
}
if (!num_weights)
exit_bad_args();
} else if (!strcmp(ctx->argv[arg_num], "hkey")) {
++arg_num;
rxfhindir_key = ctx->argv[arg_num];
if (!rxfhindir_key)
exit_bad_args();
++arg_num;
} else if (!strcmp(ctx->argv[arg_num], "default")) {
++arg_num;
rxfhindir_default = 1;
} else {
exit_bad_args();
}
}

if (rxfhindir_equal && rxfhindir_weight) {
fprintf(stderr,
"Equal and weight options are mutually exclusive\n");
return 1;
}

if (rxfhindir_equal && rxfhindir_default) {
fprintf(stderr,
"Equal and default options are mutually exclusive\n");
return 1;
}

if (rxfhindir_weight && rxfhindir_default) {
fprintf(stderr,
"Weight and default options are mutually exclusive\n");
return 1;
}

ring_count.cmd = ETHTOOL_GRXRINGS;
err = send_ioctl(ctx, &ring_count);
if (err < 0) {
perror("Cannot get RX ring count");
return 1;
}

```

```
rss_head.cmd = ETHTOOL_GRSSH;
err = send_ioctl(ctx, &rss_head);
if (err < 0 && errno == EOPNOTSUPP && !rxfhindir_key) {
return do_srxfhindir(ctx, rxfhindir_default, rxfhindir_equal,
    rxfhindir_weight, num_weights);
} else if (err < 0) {
perror("Cannot get RX flow hash indir size and key size");
return 1;
}

if (rxfhindir_key) {
err = parse_hkey(&hkey, rss_head.key_size,
    rxfhindir_key);
if (err)
return err;
}

if (rxfhindir_equal || rxfhindir_weight)
indir_bytes = rss_head.indir_size * entry_size;

rss = calloc(1, sizeof(*rss) + indir_bytes + rss_head.key_size);
if (!rss) {
perror("Cannot allocate memory for RX flow hash config");
return 1;
}
rss->cmd = ETHTOOL_SRSSH;
rss->indir_size = rss_head.indir_size;
rss->key_size = rss_head.key_size;

if (fill_indir_table(&rss->indir_size, rss->rss_config, rxfhindir_default,
    rxfhindir_equal, rxfhindir_weight, num_weights)) {
err = 1;
goto free;
}

if (hkey)
memcpy((char *)rss->rss_config + indir_bytes,
    hkey, rss->key_size);
else
rss->key_size = 0;

err = send_ioctl(ctx, rss);
if (err < 0) {
perror("Cannot set RX flow hash configuration");
err = 1;
}

free:
if (hkey)
free(hkey);

free(rss);
return err;
}
```

```

static int do_flash(struct cmd_context *ctx)
{
    char *flash_file;
    int flash_region;
    struct ethtool_flash efl;
    int err;

    if (ctx->argc < 1 || ctx->argc > 2)
        exit_bad_args();
    flash_file = ctx->argv[0];
    if (ctx->argc == 2) {
        flash_region = strtol(ctx->argv[1], NULL, 0);
        if (flash_region < 0)
            exit_bad_args();
    } else {
        flash_region = -1;
    }

    if (strlen(flash_file) > ETHTOOL_FLASH_MAX_FILENAME - 1) {
        fprintf(stdout, "Filename too long\n");
        return 99;
    }

    efl.cmd = ETHTOOL_FLASHDEV;
    strcpy(efl.data, flash_file);

    if (flash_region < 0)
        efl.region = ETHTOOL_FLASH_ALL_REGIONS;
    else
        efl.region = flash_region;

    err = send_ioctl(ctx, &efl);
    if (err < 0)
        perror("Flashing failed");

    return err;
}

static int do_permaddr(struct cmd_context *ctx)
{
    int i, err;
    struct ethtool_perm_addr *epaddr;

    epaddr = malloc(sizeof(struct ethtool_perm_addr) + MAX_ADDR_LEN);
    epaddr->cmd = ETHTOOL_GPERMADDR;
    epaddr->size = MAX_ADDR_LEN;

    err = send_ioctl(ctx, epaddr);
    if (err < 0)
        perror("Cannot read permanent address");
    else {
        printf("Permanent address:");
        for (i = 0; i < epaddr->size; i++)
            printf("%c%02x", (i == 0) ? ':' : ': ',

```

```
        epaddr->data[i]);
    printf("\n");
}
free(epaddr);

return err;
}

static bool flow_type_is_ntuple_supported(__u32 flow_type)
{
    switch (flow_type) {
    case TCP_V4_FLOW:
    case UDP_V4_FLOW:
    case SCTP_V4_FLOW:
    case AH_V4_FLOW:
    case ESP_V4_FLOW:
    case IPV4_USER_FLOW:
    case ETHER_FLOW:
    return true;
    default:
    return false;
    }
}

static int flow_spec_to_ntuple(struct ethtool_rx_flow_spec *fsp,
                             struct ethtool_rx_ntuple_flow_spec *ntuple)
{
    size_t i;

    /* verify location is not specified */
    if (fsp->location != RX_CLS_LOC_ANY)
        return -1;

    /* destination MAC address in L3/L4 rules is not supported by ntuple */
    if (fsp->flow_type & FLOW_MAC_EXT)
        return -1;

    /* verify ring cookie can transfer to action */
    if (fsp->ring_cookie > INT_MAX && fsp->ring_cookie < (u64)(-2))
        return -1;

    /* verify only one field is setting data field */
    if ((fsp->flow_type & FLOW_EXT) &&
        (fsp->m_ext.data[0] || fsp->m_ext.data[1]) &&
        fsp->m_ext.vlan_etype)
        return -1;

    /* IPv6 flow types are not supported by ntuple */
    if (!flow_type_is_ntuple_supported(fsp->flow_type & ~FLOW_EXT))
        return -1;

    /* Set entire ntuple to ~0 to guarantee all masks are set */
    memset(ntuple, ~0, sizeof(*ntuple));
}
```

```

/* set non-filter values */
ntuple->flow_type = fsp->flow_type;
ntuple->action = fsp->ring_cookie;

/*
 * Copy over header union, they are identical in layout however
 * the ntuple union contains additional padding on the end
 */
memcpy(&ntuple->h_u, &fsp->h_u, sizeof(fsp->h_u));

/*
 * The same rule mentioned above applies to the mask union. However,
 * in addition we need to invert the mask bits to match the ntuple
 * mask which is 1 for masked, versus 0 for masked as seen in nfc.
 */
memcpy(&ntuple->m_u, &fsp->m_u, sizeof(fsp->m_u));
for (i = 0; i < sizeof(fsp->m_u); i++)
    ntuple->m_u.hdata[i] ^= 0xFF;

/* copy extended fields */
if (fsp->flow_type & FLOW_EXT) {
    ntuple->vlan_tag =
        ntohs(fsp->h_ext.vlan_tci);
    ntuple->vlan_tag_mask =
        ~ntohs(fsp->m_ext.vlan_tci);
    if (fsp->m_ext.vlan_etype) {
/*
 * vlan_etype and user data are mutually exclusive
 * in ntuple configuration as they occupy the same
 * space.
 */
        if (fsp->m_ext.data[0] || fsp->m_ext.data[1])
            return -1;
        ntuple->data =
            ntohl(fsp->h_ext.vlan_etype);
        ntuple->data_mask =
            ~(u64)ntohl(fsp->m_ext.vlan_etype);
    } else {
        ntuple->data =
            (u64)ntohl(fsp->h_ext.data[0]) << 32;
        ntuple->data |=
            (u64)ntohl(fsp->h_ext.data[1]);
        ntuple->data_mask =
            (u64)ntohl(~fsp->m_ext.data[0]) << 32;
        ntuple->data_mask |=
            (u64)ntohl(~fsp->m_ext.data[1]);
    }
}

/* Mask out the extended bit, because ntuple does not know it! */
ntuple->flow_type &= ~FLOW_EXT;

return 0;
}

```

```

static int do_srxntuple(struct cmd_context *ctx,
struct ethtool_rx_flow_spec *rx_rule_fs)
{
struct ethtool_rx_ntuple ntuplecmd;
struct ethtool_value eval;
int err;

/* attempt to convert the flow classifier to an ntuple classifier */
err = flow_spec_to_ntuple(rx_rule_fs, &ntuplecmd.fs);
if (err)
return -1;

/*
 * Check to see if the flag is set for N-tuple, this allows
 * us to avoid the possible EINVAL response for the N-tuple
 * flag not being set on the device
 */
eval.cmd = ETHTOOL_GFLAGS;
err = send_ioctl(ctx, &eval);
if (err || !(eval.data & ETH_FLAG_NTUPLE))
return -1;

/* send rule via N-tuple */
ntuplecmd.cmd = ETHTOOL_SRXNTUPLE;
err = send_ioctl(ctx, &ntuplecmd);

/*
 * Display error only if response is something other than op not
 * supported. It is possible that the interface uses the network
 * flow classifier interface instead of N-tuple.
 */
if (err < 0) {
if (errno != EOPNOTSUPP)
perror("Cannot add new rule via N-tuple");
return -1;
}

return 0;
}

static int do_writewdump(struct ethtool_dump *dump, const char *dump_file)
{
int err = 0;
FILE *f;
size_t bytes;

f = fopen(dump_file, "wb+");

if (!f) {
fprintf(stderr, "Can't open file %s: %s\n",
dump_file, strerror(errno));
return 1;
}
bytes = fwrite(dump->data, 1, dump->len, f);

```

```

    if (bytes != dump->len) {
        fprintf(stderr, "Can not write all of dump data\n");
        err = 1;
    }
    if (fclose(f)) {
        fprintf(stderr, "Can't close file %s: %s\n",
            dump_file, strerror(errno));
        err = 1;
    }
    return err;
}

static int do_getfwdump(struct cmd_context *ctx)
{
    u32 dump_flag;
    char *dump_file;
    int err;
    struct ethtool_dump edata;
    struct ethtool_dump *data;

    if (ctx->argc == 2 && !strcmp(ctx->argv[0], "data")) {
        dump_flag = ETHTOOL_GET_DUMP_DATA;
        dump_file = ctx->argv[1];
    } else if (ctx->argc == 0) {
        dump_flag = 0;
        dump_file = NULL;
    } else {
        exit_bad_args();
    }

    edata.cmd = ETHTOOL_GET_DUMP_FLAG;

    err = send_ioctl(ctx, &edata);
    if (err < 0) {
        perror("Can not get dump level\n");
        return 1;
    }
    if (dump_flag != ETHTOOL_GET_DUMP_DATA) {
        fprintf(stdout, "flag: %u, version: %u, length: %u\n",
            edata.flag, edata.version, edata.len);
        return 0;
    }
    data = calloc(1, offsetof(struct ethtool_dump, data) + edata.len);
    if (!data) {
        perror("Can not allocate enough memory\n");
        return 1;
    }
    data->cmd = ETHTOOL_GET_DUMP_DATA;
    data->len = edata.len;
    err = send_ioctl(ctx, data);
    if (err < 0) {
        perror("Can not get dump data\n");
        err = 1;
        goto free;
    }
}

```



```
err = do_writewdump(data, dump_file);
free:
free(data);
return err;
}

static int do_setfdump(struct cmd_context *ctx)
{
    u32 dump_flag;
    int err;
    struct ethtool_dump dump;

    if (ctx->argc != 1)
        exit_bad_args();
    dump_flag = get_u32(ctx->argv[0], 0);

    dump.cmd = ETHTOOL_SET_DUMP;
    dump.flag = dump_flag;
    err = send_ioctl(ctx, &dump);
    if (err < 0) {
        perror("Can not set dump level\n");
        return 1;
    }
    return 0;
}

static int do_gprivflags(struct cmd_context *ctx)
{
    struct ethtool_gstrings *strings;
    struct ethtool_value flags;
    unsigned int i;
    int max_len = 0, cur_len;

    if (ctx->argc != 0)
        exit_bad_args();

    strings = get_stringset(ctx, ETH_SS_PRIV_FLAGS,
        offsetof(struct ethtool_drvinfo, n_priv_flags),
        1);
    if (!strings) {
        perror("Cannot get private flag names");
        return 1;
    }
    if (strings->len == 0) {
        fprintf(stderr, "No private flags defined\n");
        return 1;
    }
    if (strings->len > 32) {
        /* ETHTOOL_GPFLAGS can only cover 32 flags */
        fprintf(stderr, "Only showing first 32 private flags\n");
        strings->len = 32;
    }

    flags.cmd = ETHTOOL_GPFLAGS;
```

```

if (send_ioctl(ctx, &flags)) {
    perror("Cannot get private flags");
    return 1;
}

/* Find longest string and align all strings accordingly */
for (i = 0; i < strings->len; i++) {
    cur_len = strlen((const char*)strings->data +
        i * ETH_GSTRING_LEN);
    if (cur_len > max_len)
        max_len = cur_len;
}

printf("Private flags for %s:\n", ctx->devname);
for (i = 0; i < strings->len; i++)
    printf("%-*s: %s\n",
        max_len,
        (const char *)strings->data + i * ETH_GSTRING_LEN,
        (flags.data & (1U << i)) ? "on" : "off");

return 0;
}

static int do_sprivflags(struct cmd_context *ctx)
{
    struct ethtool_gstrings *strings;
    struct cmdline_info *cmdline;
    struct ethtool_value flags;
    u32 wanted_flags = 0, seen_flags = 0;
    int any_changed;
    unsigned int i;

    strings = get_stringset(ctx, ETH_SS_PRIV_FLAGS,
        offsetof(struct ethtool_drvinfo, n_priv_flags),
        1);
    if (!strings) {
        perror("Cannot get private flag names");
        return 1;
    }
    if (strings->len == 0) {
        fprintf(stderr, "No private flags defined\n");
        return 1;
    }
    if (strings->len > 32) {
        /* ETHTOOL_{G,S}PFLAGS can only cover 32 flags */
        fprintf(stderr, "Only setting first 32 private flags\n");
        strings->len = 32;
    }

    cmdline = calloc(strings->len, sizeof(*cmdline));
    if (!cmdline) {
        perror("Cannot parse arguments");
        return 1;
    }
    for (i = 0; i < strings->len; i++) {

```

```

cmdline[i].name = ((const char *)strings->data +
    i * ETH_GSTRING_LEN);
cmdline[i].type = CMDL_FLAG;
cmdline[i].wanted_val = &wanted_flags;
cmdline[i].flag_val = 1U << i;
cmdline[i].seen_val = &seen_flags;
}
parse_generic_cmdline(ctx, &any_changed, cmdline, strings->len);
free(cmdline);

flags.cmd = ETHTOOL_GPFLAGS;
if (send_ioctl(ctx, &flags)) {
    perror("Cannot get private flags");
    return 1;
}

flags.cmd = ETHTOOL_SPFLAGS;
flags.data = (flags.data & ~seen_flags) | wanted_flags;
if (send_ioctl(ctx, &flags)) {
    perror("Cannot set private flags");
    return 1;
}

return 0;
}

static int do_tsinfo(struct cmd_context *ctx)
{
    struct ethtool_ts_info info;

    if (ctx->argc != 0)
        exit_bad_args();

    fprintf(stdout, "Time stamping parameters for %s:\n", ctx->devname);
    info.cmd = ETHTOOL_GET_TS_INFO;
    if (send_ioctl(ctx, &info)) {
        perror("Cannot get device time stamping settings");
        return -1;
    }
    dump_tsinfo(&info);
    return 0;
}

static int do_getmodule(struct cmd_context *ctx)
{
    struct ethtool_modinfo modinfo;
    struct ethtool_eeprom *eeprom;
    u32 geeprom_offset = 0;
    u32 geeprom_length = -1;
    int geeprom_changed = 0;
    int geeprom_dump_raw = 0;
    int geeprom_dump_hex = 0;
    int err;

```

```

struct cmdline_info cmdline_geeprom[] = {
    { "offset", CMDL_U32, &geeprom_offset, NULL },
    { "length", CMDL_U32, &geeprom_length, NULL },
    { "raw", CMDL_BOOL, &geeprom_dump_raw, NULL },
    { "hex", CMDL_BOOL, &geeprom_dump_hex, NULL },
};

parse_generic_cmdline(ctx, &geeprom_changed,
    cmdline_geeprom, ARRAY_SIZE(cmdline_geeprom));

if (geeprom_dump_raw && geeprom_dump_hex) {
    printf("Hex and raw dump cannot be specified together\n");
    return 1;
}

modinfo.cmd = ETHTOOL_GMODULEINFO;
err = send_ioctl(ctx, &modinfo);
if (err < 0) {
    perror("Cannot get module EEPROM information");
    return 1;
}

if (geeprom_length == -1)
    geeprom_length = modinfo.eeprom_len;

if (modinfo.eeprom_len < geeprom_offset + geeprom_length)
    geeprom_length = modinfo.eeprom_len - geeprom_offset;

eeprom = calloc(1, sizeof(*eeprom)+geeprom_length);
if (!eeprom) {
    perror("Cannot allocate memory for Module EEPROM data");
    return 1;
}

eeprom->cmd = ETHTOOL_GMODULEEEPROM;
eeprom->len = geeprom_length;
eeprom->offset = geeprom_offset;
err = send_ioctl(ctx, eeprom);
if (err < 0) {
    perror("Cannot get Module EEPROM data");
    free(eeprom);
    return 1;
}

/*
 * SFF-8079 EEPROM layout contains the memory available at A0 address on
 * the PHY EEPROM.
 * SFF-8472 defines a virtual extension of the EEPROM, where the
 * microcontroller on the SFP/SFP+ generates a page at the A2 address,
 * which contains data relative to optical diagnostics.
 * The current kernel implementation returns a blob, which contains:
 * - ETH_MODULE_SFF_8079 => The A0 page only.
 * - ETH_MODULE_SFF_8472 => The A0 and A2 page concatenated.
 */
if (geeprom_dump_raw) {

```

```

fwrite(eeprom->data, 1, eeprom->len, stdout);
} else {
if (eeprom->offset != 0 ||
    (eeprom->len != modinfo.eeprom_len)) {
geeprom_dump_hex = 1;
} else if (!geeprom_dump_hex) {
switch (modinfo.type) {
#ifdef ETHTOOL_ENABLE_PRETTY_DUMP
case ETH_MODULE_SFF_8079:
sff8079_show_all(eeprom->data);
break;
case ETH_MODULE_SFF_8472:
sff8079_show_all(eeprom->data);
sff8472_show_all(eeprom->data);
break;
#endif
default:
geeprom_dump_hex = 1;
break;
}
}
if (geeprom_dump_hex)
dump_hex(stdout, eeprom->data,
    eeprom->len, eeprom->offset);
}

free(eeprom);

return 0;
}

static int do_geee(struct cmd_context *ctx)
{
struct ethtool_eee eeecmd;

if (ctx->argc != 0)
exit_bad_args();

eeecmd.cmd = ETHTOOL_GEEE;
if (send_ioctl(ctx, &eeecmd)) {
perror("Cannot get EEE settings");
return 1;
}

fprintf(stdout, "EEE Settings for %s:\n", ctx->devname);
dump_eeecmd(&eeecmd);

return 0;
}

static int do_seee(struct cmd_context *ctx)
{
int adv_c = -1, lpi_c = -1, lpi_time_c = -1, eee_c = -1;
int change = -1, change2 = 0;

```

```

struct ethtool_eee eeecmd;
struct cmdline_info cmdline_eee[] = {
    { "advertise",  CMDL_U32, &adv_c,    &eeecmd.advertised },
    { "tx-lpi",     CMDL_BOOL, &lpi_c,   &eeecmd.tx_lpi_enabled },
    { "tx-timer",   CMDL_U32, &lpi_time_c, &eeecmd.tx_lpi_timer },
    { "eee",        CMDL_BOOL, &eee_c,   &eeecmd.eee_enabled },
};

if (ctx->argc == 0)
    exit_bad_args();

parse_generic_cmdline(ctx, &change, cmdline_eee,
    ARRAY_SIZE(cmdline_eee));

eeecmd.cmd = ETHTOOL_GEEE;
if (send_ioctl(ctx, &eeecmd)) {
    perror("Cannot get EEE settings");
    return 1;
}

do_generic_set(cmdline_eee, ARRAY_SIZE(cmdline_eee), &change2);

if (change2) {

    eeecmd.cmd = ETHTOOL_SEEE;
    if (send_ioctl(ctx, &eeecmd)) {
        perror("Cannot set EEE settings");
        return 1;
    }
}

return 0;
}

#ifdef TEST_ETHTOOL
int send_ioctl(struct cmd_context *ctx, void *cmd)
{
    ctx->ifr.ifr_data = cmd;
    return ioctl(ctx->fd, SIOCETHTOOL, &ctx->ifr);
}
#endif

static int show_usage(struct cmd_context *ctx);

static const struct option {
    const char *opts;
    int want_device;
    int (*func)(struct cmd_context *);
    char *help;
    char *opthelp;
} args[] = {
    { "-s|--change", 1, do_sset, "Change generic options",
      "[ speed %d ]\n"
      "[ duplex half|full ]\n"
      "[ port tp|aui|bnc|mii|fibre ]\n"

```

```

"[ mdix auto|on|off ]\n"
"[ autoneg on|off ]\n"
"[ advertise %x ]\n"
"[ phyad %d ]\n"
"[ xcvr internal|external ]\n"
"[ wol p|u|m|b|a|g|s|d... ]\n"
"[ sopass %x:%x:%x:%x:%x:%x ]\n"
"[ msglvl %d | msglvl type on|off ... ]\n" },
{ "-a|--show-pause", 1, do_gpause, "Show pause options" },
{ "-A|--pause", 1, do_spause, "Set pause options",
"[ autoneg on|off ]\n"
"[ rx on|off ]\n"
"[ tx on|off ]\n" },
{ "-c|--show-coalesce", 1, do_gcoalesce, "Show coalesce options" },
{ "-C|--coalesce", 1, do_scoalesce, "Set coalesce options",
"[adaptive-rx on|off]\n"
"[adaptive-tx on|off]\n"
"[rx-usecs N]\n"
"[rx-frames N]\n"
"[rx-usecs-irq N]\n"
"[rx-frames-irq N]\n"
"[tx-usecs N]\n"
"[tx-frames N]\n"
"[tx-usecs-irq N]\n"
"[tx-frames-irq N]\n"
"[stats-block-usecs N]\n"
"[pkt-rate-low N]\n"
"[rx-usecs-low N]\n"
"[rx-frames-low N]\n"
"[tx-usecs-low N]\n"
"[tx-frames-low N]\n"
"[pkt-rate-high N]\n"
"[rx-usecs-high N]\n"
"[rx-frames-high N]\n"
"[tx-usecs-high N]\n"
"[tx-frames-high N]\n"
"[sample-interval N]\n" },
{ "-g|--show-ring", 1, do_gring, "Query RX/TX ring parameters" },
{ "-G|--set-ring", 1, do_sring, "Set RX/TX ring parameters",
"[ rx N ]\n"
"[ rx-mini N ]\n"
"[ rx-jumbo N ]\n"
"[ tx N ]\n" },
{ "-k|--show-features|--show-offload", 1, do_gfeatures,
"Get state of protocol offload and other features" },
{ "-K|--features|--offload", 1, do_sfeatures,
"Set protocol offload and other features",
"FEATURE on|off ...\n" },
{ "-i|--driver", 1, do_gdrv, "Show driver information" },
{ "-d|--register-dump", 1, do_gregs, "Do a register dump",
"[ raw on|off ]\n"
"[ file FILENAME ]\n" },
{ "-e|--eeprom-dump", 1, do_geeprom, "Do a EEPROM dump",
"[ raw on|off ]\n"

```

```

    "[ offset N ]\n"
    "[ length N ]\n" },
{ "-E|--change-EEPROM", 1, do_seeprom,
  "Change bytes in device EEPROM",
  "[ magic N ]\n"
  "[ offset N ]\n"
  "[ length N ]\n"
  "[ value N ]\n" },
{ "-r|--negotiate", 1, do_nway_rst, "Restart N-WAY negotiation" },
{ "-p|--identify", 1, do_phys_id,
  "Show visible port identification (e.g. blinking)",
  "          [ TIME-IN-SECONDS ]\n" },
{ "-t|--test", 1, do_test, "Execute adapter self test",
  "          [ online | offline | external_lb ]\n" },
{ "-S|--statistics", 1, do_gnicstats, "Show adapter statistics" },
{ "--phy-statistics", 1, do_gphystats,
  "Show phy statistics" },
{ "-n|-u|--show-nfc|--show-ntuple", 1, do_grxclass,
  "Show Rx network flow classification options or rules",
  "[ rx-flow-hash tcp4|udp4|ah4|esp4|sctp4|"
  "tcp6|udp6|ah6|esp6|sctp6 ]\n"
  " rule %d ]\n" },
{ "-N|-U|--config-nfc|--config-ntuple", 1, do_srxclass,
  "Configure Rx network flow classification options or rules",
  "rx-flow-hash tcp4|udp4|ah4|esp4|sctp4|"
  "tcp6|udp6|ah6|esp6|sctp6 m|v|t|s|d|f|n|r... ]\n"
  "flow-type ether|ip4|tcp4|udp4|sctp4|ah4|esp4|"
  "ip6|tcp6|udp6|ah6|esp6|sctp6\n"
  "[ src %x:%x:%x:%x:%x:%x [m %x:%x:%x:%x:%x:%x] ]\n"
  "[ dst %x:%x:%x:%x:%x:%x [m %x:%x:%x:%x:%x:%x] ]\n"
  "[ proto %d [m %x] ]\n"
  "[ src-ip IP-ADDRESS [m IP-ADDRESS] ]\n"
  "[ dst-ip IP-ADDRESS [m IP-ADDRESS] ]\n"
  "[ tos %d [m %x] ]\n"
  "[ tclass %d [m %x] ]\n"
  "[ l4proto %d [m %x] ]\n"
  "[ src-port %d [m %x] ]\n"
  "[ dst-port %d [m %x] ]\n"
  "[ spi %d [m %x] ]\n"
  "[ vlan-etype %x [m %x] ]\n"
  "[ vlan %x [m %x] ]\n"
  "[ user-def %x [m %x] ]\n"
  "[ dst-mac %x:%x:%x:%x:%x:%x [m %x:%x:%x:%x:%x:%x] ]\n"
  "[ action %d ]\n"
  "[ loc %d]] ]\n"
  "delete %d\n" },
{ "-T|--show-time-stamping", 1, do_tsinfo,
  "Show time stamping capabilities" },
{ "-x|--show-rxfh-indir|--show-rxfh", 1, do_grxfh,
  "Show Rx flow hash indirection and/or hash key" },
{ "-X|--set-rxfh-indir|--rxfh", 1, do_srxfh,
  "Set Rx flow hash indirection and/or hash key",
  "[ equal N | weight W0 W1 ... ]\n"
  "[ hkey %x:%x:%x:%x:%x:.... ]\n" },
{ "-f|--flash", 1, do_flash,

```



```

    "Flash firmware image from the specified file to a region on the device",
    "    FILENAME [ REGION-NUMBER-TO-FLASH ]\n" },
{ "-P|--show-permaddr", 1, do_permaddr,
  "Show permanent hardware address" },
{ "-w|--get-dump", 1, do_getfwdump,
  "Get dump flag, data",
  "[ data FILENAME ]\n" },
{ "-W|--set-dump", 1, do_setfwdump,
  "Set dump flag of the device",
  "N\n"},
{ "-l|--show-channels", 1, do_gchannels, "Query Channels" },
{ "-L|--set-channels", 1, do_schannels, "Set Channels",
  "    [ rx N ]\n"
  "    [ tx N ]\n"
  "    [ other N ]\n"
  "    [ combined N ]\n" },
{ "--show-priv-flags", 1, do_gprivflags, "Query private flags" },
{ "--set-priv-flags", 1, do_sprivflags, "Set private flags",
  "FLAG on/off ...\n" },
{ "-m|--dump-module-EEPROM|--module-info", 1, do_getmodule,
  "Query/Decode Module EEPROM information and optical diagnostics if available",
  "[ raw on/off ]\n"
  "[ hex on/off ]\n"
  "[ offset N ]\n"
  "[ length N ]\n" },
{ "--show-eee", 1, do_geee, "Show EEE settings"},
{ "--set-eee", 1, do_seee, "Set EEE settings",
  "[ eee on/off ]\n"
  "[ advertise %x ]\n"
  "[ tx-lpi on/off ]\n"
  "[ tx-timer %d ]\n"},
{ "-h|--help", 0, show_usage, "Show this help" },
{ "--version", 0, do_version, "Show version number" },
{}
};

static int show_usage(struct cmd_context *ctx)
{
    int i;

    /* ethtool -h */
    fprintf(stdout, PACKAGE " version " VERSION "\n");
    fprintf(stdout,
    "Usage:\n"
    "    ethtool DEVNAME\t"
    "Display standard information about device\n");
    for (i = 0; args[i].opts; i++) {
        fputs("    ethtool ", stdout);
        fprintf(stdout, "%s %s\t%s\n",
        args[i].opts,
        args[i].want_device ? "DEVNAME" : "\t",
        args[i].help);
        if (args[i].opthelp)
            fputs(args[i].opthelp, stdout);
    }
}

```

```

    }

    return 0;
}

int main(int argc, char **argp)
{
    int (*func)(struct cmd_context *);
    int want_device;
    struct cmd_context ctx;
    int k;

    init_global_link_mode_masks();

    /* Skip command name */
    argp++;
    argc--;

    /* First argument must be either a valid option or a device
     * name to get settings for (which we don't expect to begin
     * with '-').
     */
    if (argc == 0)
        exit_bad_args();
    for (k = 0; args[k].opts; k++) {
        const char *opt;
        size_t len;
        opt = args[k].opts;
        for (;;) {
            len = strcspn(opt, "|");
            if (strncmp(*argp, opt, len) == 0 &&
                (*argp)[len] == 0) {
                argp++;
                argc--;
                func = args[k].func;
                want_device = args[k].want_device;
                goto opt_found;
            }
            if (opt[len] == 0)
                break;
            opt += len + 1;
        }
    }
    if ((*argp)[0] == '-')
        exit_bad_args();
    func = do_gset;
    want_device = 1;

opt_found:
    if (want_device) {
        ctx.devname = *argp++;
        argc--;

        if (ctx.devname == NULL)
            exit_bad_args();
    }

```

```
if (strlen(ctx.devname) >= IFNAMSIZ)
    exit_bad_args();

/* Setup our control structures. */
memset(&ctx.ifr, 0, sizeof(ctx.ifr));
strcpy(ctx.ifr.ifr_name, ctx.devname);

/* Open control socket. */
ctx.fd = socket(AF_INET, SOCK_DGRAM, 0);
if (ctx.fd < 0)
    ctx.fd = socket(AF_NETLINK, SOCK_RAW, NETLINK_GENERIC);
if (ctx.fd < 0) {
    perror("Cannot get control socket");
    return 70;
}
} else {
    ctx.fd = -1;
}

ctx.argc = argc;
ctx.argp = argp;

return func(&ctx);
}
```

1.31 eudev

1.31.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.32 expat

1.32.1 Available under license

Copyright (c) 1998-2000 Thai Open Source Software Center Ltd and Clark Cooper
Copyright (c) 2001-2016 Expat maintainers

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.33 expect

1.33.1 Available under license

Expect

Written by: Don Libes, libes at nist.gov, NIST

Design and implementation of this program was paid for by U.S. tax dollars. Therefore it is public domain. However, the author and NIST would appreciate credit if this program or parts of it are used.

1.34 file

1.34.1 Available under license

\$File: LEGAL.NOTICE,v 1.15 2006/05/03 18:48:33 christos Exp \$
Copyright (c) Ian F. Darwin 1986, 1987, 1989, 1990, 1991, 1992, 1994, 1995.
Software written by Ian F. Darwin and others;
maintained 1994- Christos Zoulas.

This software is not subject to any export provision of the United States Department of Commerce, and may be exported to any country or planet.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice immediately at the beginning of the file, without modification, this list of conditions, and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.35 findutils

1.35.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as

part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may

otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.36 flex

1.36.1 Available under license

Flex carries the copyright used for BSD software, slightly modified because it originated at the Lawrence Berkeley (not Livermore!) Laboratory, which operates under a contract with the Department of Energy:

Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006, 2007 The Flex Project.

Copyright (c) 1990, 1997 The Regents of the University of California.
All rights reserved.

This code is derived from software contributed to Berkeley by
Vern Paxson.

The United States Government has rights in this work pursuant
to contract no. DE-AC03-76SF00098 between the United States
Department of Energy and the University of California.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the University nor the names of its contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.

This basically says "do whatever you please with this software except
remove this notice or take advantage of the University's (or the flex
authors') name".

Note that the "flex.skl" scanner skeleton carries no copyright notice.
You are free to do whatever you please with scanners generated using flex;
for them, you are not even bound by the above copyright.

1.37 gawk

1.37.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.38 gdbm

1.38.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007, 2011 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.39 gettext

1.39.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.40 glib-2.0

1.40.1 Available under license

This work may be reproduced and distributed in whole or in part, in any medium, physical or electronic, so as long as this copyright notice remains intact and unchanged on all copies. Commercial redistribution is permitted and encouraged, but you may not redistribute, in whole or in part, under terms more restrictive than those under which you received it. If you redistribute a modified or translated version of this work, you must also make the source code to the modified or translated version available in electronic form without charge. However, mere aggregation as part of a larger work shall not count as a modification for this purpose.

All code examples in this work are placed into the public domain, and may be used, modified and redistributed without restriction.

BECAUSE THIS WORK IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE WORK, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE WORK "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SHOULD THE WORK PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE WORK AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE WORK, EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is

analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation

and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you

distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the

library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy

from a designated place, offer equivalent access to copy the above specified materials from the same place.

d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and

conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307 USA.

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!
PCRE LICENCE

Please see the file LICENCE in the PCRE distribution for licensing details.

End

```
/* GLIB - Library of useful routines for C programming
 * Copyright (C) 1995-1997 Peter Mattis, Spencer Kimball and Josh MacDonald
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, see <http://www.gnu.org/licenses/>.
 */

/*
 * Modified by the GLib Team and others 1997-2000. See the AUTHORS
 * file for a list of people on the GLib Team. See the ChangeLog
 * files for a list of changes. These files are distributed with
```

```
* GLib at ftp://ftp.gtk.org/pub/gtk/.  
*/
```

```
#ifndef __G_LIB_H__  
#define __G_LIB_H__
```

```
#define __GLIB_H_INSIDE__
```

```
#include <glib/galloca.h>  
#include <glib/garray.h>  
#include <glib/gasyncqueue.h>  
#include <glib/gatomic.h>  
#include <glib/gbacktrace.h>  
#include <glib/gbase64.h>  
#include <glib/gbitlock.h>  
#include <glib/gbookmarkfile.h>  
#include <glib/gbytes.h>  
#include <glib/gcharset.h>  
#include <glib/gchecksum.h>  
#include <glib/gconvert.h>  
#include <glib/gdataset.h>  
#include <glib/gdate.h>  
#include <glib/gdatetime.h>  
#include <glib/gdir.h>  
#include <glib/genviron.h>  
#include <glib/gerror.h>  
#include <glib/gfileutils.h>  
#include <glib/ggettext.h>  
#include <glib/ghash.h>  
#include <glib/ghmac.h>  
#include <glib/ghook.h>  
#include <glib/ghostutils.h>  
#include <glib/giochannel.h>  
#include <glib/gkeyfile.h>  
#include <glib/glist.h>  
#include <glib/gmacros.h>  
#include <glib/gmain.h>  
#include <glib/gmappedfile.h>  
#include <glib/gmarkup.h>  
#include <glib/gmem.h>  
#include <glib/gmessages.h>  
#include <glib/gnode.h>  
#include <glib/goption.h>  
#include <glib/gpattern.h>  
#include <glib/gpoll.h>  
#include <glib/gprimes.h>  
#include <glib/gqsort.h>  
#include <glib/gquark.h>  
#include <glib/gqueue.h>  
#include <glib/grand.h>  
#include <glib/gregex.h>  
#include <glib/gscanner.h>  
#include <glib/gsequence.h>  
#include <glib/gshell.h>
```

```

#include <glib/gslice.h>
#include <glib/gslist.h>
#include <glib/gspawn.h>
#include <glib/gstrfuncs.h>
#include <glib/gstring.h>
#include <glib/gstringchunk.h>
#include <glib/gtestutils.h>
#include <glib/gthread.h>
#include <glib/gthreadpool.h>
#include <glib/gtimer.h>
#include <glib/gtimezone.h>
#include <glib/gtrashstack.h>
#include <glib/gtree.h>
#include <glib/gtypes.h>
#include <glib/gunicode.h>
#include <glib/gurifuncs.h>
#include <glib/gutils.h>
#include <glib/gvarianttype.h>
#include <glib/gvariant.h>
#include <glib/gversion.h>
#include <glib/gversionmacros.h>
#ifdef G_PLATFORM_WIN32
#include <glib/gwin32.h>
#endif

#ifndef G_DISABLE_DEPRECATED
#include <glib/deprecated/gallocator.h>
#include <glib/deprecated/gcache.h>
#include <glib/deprecated/gcompletion.h>
#include <glib/deprecated/gmain.h>
#include <glib/deprecated/grel.h>
#include <glib/deprecated/gthread.h>
#endif /* G_DISABLE_DEPRECATED */

#include <glib/glib-autocleanups.h>

#undef __GLIB_H_INSIDE__

#endif /* __G_LIB_H__ */
/* GMODULE - GLIB wrapper code for dynamic module loading
 * Copyright (C) 1998 Tim Janik
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, see <http://www.gnu.org/licenses/>.
 */

```

```

/*
 * Modified by the GLib Team and others 1997-2000. See the AUTHORS
 * file for a list of people on the GLib Team. See the ChangeLog
 * files for a list of changes. These files are distributed with
 * GLib at ftp://ftp.gtk.org/pub/gtk/.
 */

#ifndef __GMODULE_H__
#define __GMODULE_H__

#include <glib.h>

G_BEGIN_DECLS

/* exporting and importing functions, this is special cased
 * to feature Windows dll stubs.
 */
#define G_MODULE_IMPORT extern
#ifdef G_PLATFORM_WIN32
# define G_MODULE_EXPORT __declspec(dllexport)
#else /* !G_PLATFORM_WIN32 */
# define G_MODULE_EXPORT
#endif /* !G_PLATFORM_WIN32 */

typedef enum
{
    G_MODULE_BIND_LAZY = 1 << 0,
    G_MODULE_BIND_LOCAL = 1 << 1,
    G_MODULE_BIND_MASK = 0x03
} GModuleFlags;

typedef struct _GModule GModule;
typedef const gchar* (*GModuleCheckInit) (GModule* module);
typedef void (*GModuleUnload) (GModule* module);

/* return TRUE if dynamic module loading is supported */
GLIB_AVAILABLE_IN_ALL
gboolean g_module_supported (void) G_GNUC_CONST;

/* open a module 'file_name' and return handle, which is NULL on error */
GLIB_AVAILABLE_IN_ALL
GModule* g_module_open (const gchar *file_name,
                        GModuleFlags flags);

/* close a previously opened module, returns TRUE on success */
GLIB_AVAILABLE_IN_ALL
gboolean g_module_close (GModule *module);

/* make a module resident so g_module_close on it will be ignored */
GLIB_AVAILABLE_IN_ALL
void g_module_make_resident (GModule *module);

/* query the last module error as a string */

```

```

GLIB_AVAILABLE_IN_ALL
const gchar *      g_module_error      (void);

/* retrieve a symbol pointer from 'module', returns TRUE on success */
GLIB_AVAILABLE_IN_ALL
gboolean          g_module_symbol      (GModule *module,
    const gchar *symbol_name,
    gpointer *symbol);

/* retrieve the file name from an existing module */
GLIB_AVAILABLE_IN_ALL
const gchar *      g_module_name        (GModule *module);

/* Build the actual file name containing a module. 'directory' is the
 * directory where the module file is supposed to be, or NULL or empty
 * in which case it should either be in the current directory or, on
 * some operating systems, in some standard place, for instance on the
 * PATH. Hence, to be absolutely sure to get the correct module,
 * always pass in a directory. The file name consists of the directory,
 * if supplied, and 'module_name' suitably decorated according to
 * the operating system's conventions (for instance lib*.so or *.dll).
 *
 * No checks are made that the file exists, or is of correct type.
 */
GLIB_AVAILABLE_IN_ALL
gchar*            g_module_build_path  (const gchar *directory,
    const gchar *module_name);

#ifdef __GTK_DOC_IGNORE__
#ifdef G_OS_WIN32
#define g_module_open g_module_open_utf8
#define g_module_name g_module_name_utf8

GLIB_AVAILABLE_IN_ALL
GModule * g_module_open_utf8 (const gchar *file_name,
    GModuleFlags flags);
GLIB_AVAILABLE_IN_ALL
const gchar *g_module_name_utf8 (GModule *module);
#endif
#endif

G_END_DECLS

#endif /* __GMODULE_H__ */
/*****
 * Perl-Compatible Regular Expressions
 *****/

/* This is the public header file for the PCRE library, to be #included by
applications that call the PCRE functions.

```

Copyright (c) 1997-2012 University of Cambridge

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the University of Cambridge nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

*/

```
#ifndef _PCRE_H
#define _PCRE_H
```

```
/* The current PCRE version information. */
```

```
#define PCRE_MAJOR      8
#define PCRE_MINOR      31
#define PCRE_PRERELEASE
#define PCRE_DATE        2012-07-06
```

```
/* When an application links to a PCRE DLL in Windows, the symbols that are
imported have to be identified as such. When building PCRE, the appropriate
export setting is defined in pcre_internal.h, which includes this file. So we
don't change existing definitions of PCRE_EXP_DECL and PCRECPP_EXP_DECL. */
```

```
#if defined(_WIN32) && !defined(PCRE_STATIC)
#  ifndef PCRE_EXP_DECL
#    define PCRE_EXP_DECL extern __declspec(dllimport)
#  endif
#  ifdef __cplusplus
#    ifndef PCRECPP_EXP_DECL
#      define PCRECPP_EXP_DECL extern __declspec(dllimport)
#    endif
#    ifndef PCRECPP_EXP_DEFN
#      define PCRECPP_EXP_DEFN __declspec(dllimport)
#    endif
#  endif
#endif
```

```

# endif
#endif

/* By default, we use the standard "extern" declarations. */

#ifndef PCRE_EXP_DECL
# ifdef __cplusplus
#   define PCRE_EXP_DECL extern "C"
# else
#   define PCRE_EXP_DECL extern
# endif
#endif

#ifdef __cplusplus
# ifndef PCRECPP_EXP_DECL
#   define PCRECPP_EXP_DECL extern
# endif
# ifndef PCRECPP_EXP_DEFN
#   define PCRECPP_EXP_DEFN
# endif
#endif

/* Have to include stdlib.h in order to ensure that size_t is defined;
it is needed here for malloc. */

#include <stdlib.h>

/* Allow for C++ users */

#ifdef __cplusplus
extern "C" {
#endif

/* Options. Some are compile-time only, some are run-time only, and some are
both, so we keep them all distinct. However, almost all the bits in the options
word are now used. In the long run, we may have to re-use some of the
compile-time only bits for runtime options, or vice versa. In the comments
below, "compile", "exec", and "DFA exec" mean that the option is permitted to
be set for those functions; "used in" means that an option may be set only for
compile, but is subsequently referenced in exec and/or DFA exec. Any of the
compile-time options may be inspected during studying (and therefore JIT
compiling). */

#define PCRE_CASELESS      0x00000001 /* Compile */
#define PCRE_MULTILINE     0x00000002 /* Compile */
#define PCRE_DOTALL       0x00000004 /* Compile */
#define PCRE_EXTENDED     0x00000008 /* Compile */
#define PCRE_ANCHORED     0x00000010 /* Compile, exec, DFA exec */
#define PCRE_DOLLAR_ENDONLY 0x00000020 /* Compile, used in exec, DFA exec */
#define PCRE_EXTRA        0x00000040 /* Compile */
#define PCRE_NOTBOL       0x00000080 /* Exec, DFA exec */
#define PCRE_NOTEOL       0x00000100 /* Exec, DFA exec */
#define PCRE_UNGREEDY     0x00000200 /* Compile */
#define PCRE_NOTEMPTY     0x00000400 /* Exec, DFA exec */
/* The next two are also used in exec and DFA exec */

```

```

#define PCRE_UTF8          0x00000800 /* Compile (same as PCRE_UTF16) */
#define PCRE_UTF16         0x00000800 /* Compile (same as PCRE_UTF8) */
#define PCRE_NO_AUTO_CAPTURE 0x00001000 /* Compile */
/* The next two are also used in exec and DFA exec */
#define PCRE_NO_UTF8_CHECK 0x00002000 /* Compile (same as PCRE_NO_UTF16_CHECK) */
#define PCRE_NO_UTF16_CHECK 0x00002000 /* Compile (same as PCRE_NO_UTF8_CHECK) */
#define PCRE_AUTO_CALLOUT 0x00004000 /* Compile */
#define PCRE_PARTIAL_SOFT 0x00008000 /* Exec, DFA exec */
#define PCRE_PARTIAL      0x00008000 /* Backwards compatible synonym */
#define PCRE_DFA_SHORTEST 0x00010000 /* DFA exec */
#define PCRE_DFA_RESTART 0x00020000 /* DFA exec */
#define PCRE_FIRSTLINE    0x00040000 /* Compile, used in exec, DFA exec */
#define PCRE_DUPNAMES     0x00080000 /* Compile */
#define PCRE_NEWLINE_CR   0x00100000 /* Compile, exec, DFA exec */
#define PCRE_NEWLINE_LF   0x00200000 /* Compile, exec, DFA exec */
#define PCRE_NEWLINE_CRLF 0x00300000 /* Compile, exec, DFA exec */
#define PCRE_NEWLINE_ANY 0x00400000 /* Compile, exec, DFA exec */
#define PCRE_NEWLINE_ANYCRLF 0x00500000 /* Compile, exec, DFA exec */
#define PCRE_BSR_ANYCRLF 0x00800000 /* Compile, exec, DFA exec */
#define PCRE_BSR_UNICODE 0x01000000 /* Compile, exec, DFA exec */
#define PCRE_JAVASCRIPT_COMPAT 0x02000000 /* Compile, used in exec */
#define PCRE_NO_START_OPTIMIZE 0x04000000 /* Compile, exec, DFA exec */
#define PCRE_NO_START_OPTIMISE 0x04000000 /* Synonym */
#define PCRE_PARTIAL_HARD 0x08000000 /* Exec, DFA exec */
#define PCRE_NOTEMPTY_ATSTART 0x10000000 /* Exec, DFA exec */
#define PCRE_UCP          0x20000000 /* Compile, used in exec, DFA exec */

/* Exec-time and get/set-time error codes */

#define PCRE_ERROR_NOMATCH (-1)
#define PCRE_ERROR_NULL (-2)
#define PCRE_ERROR_BADOPTION (-3)
#define PCRE_ERROR_BADMAGIC (-4)
#define PCRE_ERROR_UNKNOWN_OPCODE (-5)
#define PCRE_ERROR_UNKNOWN_NODE (-5) /* For backward compatibility */
#define PCRE_ERROR_NOMEMORY (-6)
#define PCRE_ERROR_NOSUBSTRING (-7)
#define PCRE_ERROR_MATCHLIMIT (-8)
#define PCRE_ERROR_CALLOUT (-9) /* Never used by PCRE itself */
#define PCRE_ERROR_BADUTF8 (-10) /* Same for 8/16 */
#define PCRE_ERROR_BADUTF16 (-10) /* Same for 8/16 */
#define PCRE_ERROR_BADUTF8_OFFSET (-11) /* Same for 8/16 */
#define PCRE_ERROR_BADUTF16_OFFSET (-11) /* Same for 8/16 */
#define PCRE_ERROR_PARTIAL (-12)
#define PCRE_ERROR_BADPARTIAL (-13)
#define PCRE_ERROR_INTERNAL (-14)
#define PCRE_ERROR_BADCOUNT (-15)
#define PCRE_ERROR_DFA_UITEM (-16)
#define PCRE_ERROR_DFA_UCOND (-17)
#define PCRE_ERROR_DFA_UMLIMIT (-18)
#define PCRE_ERROR_DFA_WSSIZE (-19)
#define PCRE_ERROR_DFA_RECURSE (-20)
#define PCRE_ERROR_RECURSIONLIMIT (-21)
#define PCRE_ERROR_NULLWSLIMIT (-22) /* No longer actually used */

```

```
#define PCRE_ERROR_BADNEWLINE    (-23)
#define PCRE_ERROR_BADOFFSET    (-24)
#define PCRE_ERROR_SHORTUTF8    (-25)
#define PCRE_ERROR_SHORTUTF16   (-25) /* Same for 8/16 */
#define PCRE_ERROR_RECURSELOOP  (-26)
#define PCRE_ERROR_JIT_STACKLIMIT (-27)
#define PCRE_ERROR_BADMODE      (-28)
#define PCRE_ERROR_BADENDIANNESS (-29)
#define PCRE_ERROR_DFA_BADRESTART (-30)
```

/* Specific error codes for UTF-8 validity checks */

```
#define PCRE_UTF8_ERR0      0
#define PCRE_UTF8_ERR1      1
#define PCRE_UTF8_ERR2      2
#define PCRE_UTF8_ERR3      3
#define PCRE_UTF8_ERR4      4
#define PCRE_UTF8_ERR5      5
#define PCRE_UTF8_ERR6      6
#define PCRE_UTF8_ERR7      7
#define PCRE_UTF8_ERR8      8
#define PCRE_UTF8_ERR9      9
#define PCRE_UTF8_ERR10     10
#define PCRE_UTF8_ERR11     11
#define PCRE_UTF8_ERR12     12
#define PCRE_UTF8_ERR13     13
#define PCRE_UTF8_ERR14     14
#define PCRE_UTF8_ERR15     15
#define PCRE_UTF8_ERR16     16
#define PCRE_UTF8_ERR17     17
#define PCRE_UTF8_ERR18     18
#define PCRE_UTF8_ERR19     19
#define PCRE_UTF8_ERR20     20
#define PCRE_UTF8_ERR21     21
```

/* Specific error codes for UTF-16 validity checks */

```
#define PCRE_UTF16_ERR0      0
#define PCRE_UTF16_ERR1      1
#define PCRE_UTF16_ERR2      2
#define PCRE_UTF16_ERR3      3
#define PCRE_UTF16_ERR4      4
```

/* Request types for pcre_fullinfo() */

```
#define PCRE_INFO_OPTIONS      0
#define PCRE_INFO_SIZE         1
#define PCRE_INFO_CAPTURECOUNT 2
#define PCRE_INFO_BACKREFMAX   3
#define PCRE_INFO_FIRSTBYTE     4
#define PCRE_INFO_FIRSTCHAR     4 /* For backwards compatibility */
#define PCRE_INFO_FIRSTTABLE     5
#define PCRE_INFO_LASTLITERAL   6
#define PCRE_INFO_NAMEENTRYSIZE 7
#define PCRE_INFO_NAMECOUNT    8
```

```

#define PCRE_INFO_NAMETABLE      9
#define PCRE_INFO_STUDYSIZE     10
#define PCRE_INFO_DEFAULT_TABLES 11
#define PCRE_INFO_OKPARTIAL     12
#define PCRE_INFO_JCHANGED      13
#define PCRE_INFO_HASCORRLF     14
#define PCRE_INFO_MINLENGTH     15
#define PCRE_INFO_JIT           16
#define PCRE_INFO_JITSIZE       17
#define PCRE_INFO_MAXLOOKBEHIND 18

/* Request types for pcre_config(). Do not re-arrange, in order to remain
compatible. */

#define PCRE_CONFIG_UTF8          0
#define PCRE_CONFIG_NEWLINE      1
#define PCRE_CONFIG_LINK_SIZE    2
#define PCRE_CONFIG_POSIX_MALLOC_THRESHOLD 3
#define PCRE_CONFIG_MATCH_LIMIT  4
#define PCRE_CONFIG_STACKRECURSE 5
#define PCRE_CONFIG_UNICODE_PROPERTIES 6
#define PCRE_CONFIG_MATCH_LIMIT_RECURSION 7
#define PCRE_CONFIG_BSR          8
#define PCRE_CONFIG_JIT          9
#define PCRE_CONFIG_UTF16       10
#define PCRE_CONFIG_JITTARGET   11

/* Request types for pcre_study(). Do not re-arrange, in order to remain
compatible. */

#define PCRE_STUDY_JIT_COMPILE      0x0001
#define PCRE_STUDY_JIT_PARTIAL_SOFT_COMPILE 0x0002
#define PCRE_STUDY_JIT_PARTIAL_HARD_COMPILE 0x0004

/* Bit flags for the pcre[16]_extra structure. Do not re-arrange or redefine
these bits, just add new ones on the end, in order to remain compatible. */

#define PCRE_EXTRA_STUDY_DATA      0x0001
#define PCRE_EXTRA_MATCH_LIMIT     0x0002
#define PCRE_EXTRA_CALLOUT_DATA    0x0004
#define PCRE_EXTRA_TABLES          0x0008
#define PCRE_EXTRA_MATCH_LIMIT_RECURSION 0x0010
#define PCRE_EXTRA_MARK            0x0020
#define PCRE_EXTRA_EXECUTABLE_JIT 0x0040

/* Types */

struct real_pcre;          /* declaration; the definition is private */
typedef struct real_pcre pcre;

struct real_pcre16;        /* declaration; the definition is private */
typedef struct real_pcre16 pcre16;

struct real_pcre_jit_stack; /* declaration; the definition is private */

```

```

typedef struct real_pcre_jit_stack pcre_jit_stack;

struct real_pcre16_jit_stack; /* declaration; the definition is private */
typedef struct real_pcre16_jit_stack pcre16_jit_stack;

/* If PCRE is compiled with 16 bit character support, PCRE_UCHAR16 must contain
a 16 bit wide signed data type. Otherwise it can be a dummy data type since
pcre16 functions are not implemented. There is a check for this in pcre_internal.h. */
#ifndef PCRE_UCHAR16
#define PCRE_UCHAR16 unsigned short
#endif

#ifndef PCRE_SPTR16
#define PCRE_SPTR16 const PCRE_UCHAR16 *
#endif

/* When PCRE is compiled as a C++ library, the subject pointer type can be
replaced with a custom type. For conventional use, the public interface is a
const char *. */

#ifndef PCRE_SPTR
#define PCRE_SPTR const char *
#endif

/* The structure for passing additional data to pcre_exec(). This is defined in
such a way as to be extensible. Always add new fields at the end, in order to
remain compatible. */

typedef struct pcre_extra {
    unsigned long int flags; /* Bits for which fields are set */
    void *study_data; /* Opaque data from pcre_study() */
    unsigned long int match_limit; /* Maximum number of calls to match() */
    void *callout_data; /* Data passed back in callouts */
    const unsigned char *tables; /* Pointer to character tables */
    unsigned long int match_limit_recursion; /* Max recursive calls to match() */
    unsigned char **mark; /* For passing back a mark pointer */
    void *executable_jit; /* Contains a pointer to a compiled jit code */
} pcre_extra;

/* Same structure as above, but with 16 bit char pointers. */

typedef struct pcre16_extra {
    unsigned long int flags; /* Bits for which fields are set */
    void *study_data; /* Opaque data from pcre_study() */
    unsigned long int match_limit; /* Maximum number of calls to match() */
    void *callout_data; /* Data passed back in callouts */
    const unsigned char *tables; /* Pointer to character tables */
    unsigned long int match_limit_recursion; /* Max recursive calls to match() */
    PCRE_UCHAR16 **mark; /* For passing back a mark pointer */
    void *executable_jit; /* Contains a pointer to a compiled jit code */
} pcre16_extra;

/* The structure for passing out data via the pcre_callout_function. We use a
structure so that new fields can be added on the end in future versions,
without changing the API of the function, thereby allowing old clients to work

```

without modification. */

```
typedef struct pcre_callout_block {
    int      version;      /* Identifies version of block */
    /* ----- Version 0 ----- */
    int      callout_number; /* Number compiled into pattern */
    int      *offset_vector; /* The offset vector */
    PCRE_SPTR subject;      /* The subject being matched */
    int      subject_length; /* The length of the subject */
    int      start_match;    /* Offset to start of this match attempt */
    int      current_position; /* Where we currently are in the subject */
    int      capture_top;    /* Max current capture */
    int      capture_last;   /* Most recently closed capture */
    void     *callout_data;  /* Data passed in with the call */
    /* ----- Added for Version 1 ----- */
    int      pattern_position; /* Offset to next item in the pattern */
    int      next_item_length; /* Length of next item in the pattern */
    /* ----- Added for Version 2 ----- */
    const unsigned char *mark; /* Pointer to current mark or NULL */
    /* ----- */
} pcre_callout_block;
```

/* Same structure as above, but with 16 bit char pointers. */

```
typedef struct pcre16_callout_block {
    int      version;      /* Identifies version of block */
    /* ----- Version 0 ----- */
    int      callout_number; /* Number compiled into pattern */
    int      *offset_vector; /* The offset vector */
    PCRE_SPTR16 subject;    /* The subject being matched */
    int      subject_length; /* The length of the subject */
    int      start_match;    /* Offset to start of this match attempt */
    int      current_position; /* Where we currently are in the subject */
    int      capture_top;    /* Max current capture */
    int      capture_last;   /* Most recently closed capture */
    void     *callout_data;  /* Data passed in with the call */
    /* ----- Added for Version 1 ----- */
    int      pattern_position; /* Offset to next item in the pattern */
    int      next_item_length; /* Length of next item in the pattern */
    /* ----- Added for Version 2 ----- */
    const PCRE_UCHAR16 *mark; /* Pointer to current mark or NULL */
    /* ----- */
} pcre16_callout_block;
```

/* Indirection for store get and free functions. These can be set to alternative malloc/free functions if required. Special ones are used in the non-recursive case for "frames". There is also an optional callout function that is triggered by the (?) regex item. For Virtual Pascal, these definitions have to take another form. */

```
#ifndef VPCOMPAT
PCRE_EXP_DECL void *(*pcre_malloc)(size_t);
PCRE_EXP_DECL void (*pcre_free)(void *);
PCRE_EXP_DECL void *(*pcre_stack_malloc)(size_t);
```

```

PCRE_EXP_DECL void (*pcre_stack_free)(void *);
PCRE_EXP_DECL int (*pcre_callout)(pcre_callout_block *);

PCRE_EXP_DECL void (*pcre16_malloc)(size_t);
PCRE_EXP_DECL void (*pcre16_free)(void *);
PCRE_EXP_DECL void (*pcre16_stack_malloc)(size_t);
PCRE_EXP_DECL void (*pcre16_stack_free)(void *);
PCRE_EXP_DECL int (*pcre16_callout)(pcre16_callout_block *);
#else /* VPCOMPAT */
PCRE_EXP_DECL void *pcre_malloc(size_t);
PCRE_EXP_DECL void pcre_free(void *);
PCRE_EXP_DECL void *pcre_stack_malloc(size_t);
PCRE_EXP_DECL void pcre_stack_free(void *);
PCRE_EXP_DECL int pcre_callout(pcre_callout_block *);

PCRE_EXP_DECL void *pcre16_malloc(size_t);
PCRE_EXP_DECL void pcre16_free(void *);
PCRE_EXP_DECL void *pcre16_stack_malloc(size_t);
PCRE_EXP_DECL void pcre16_stack_free(void *);
PCRE_EXP_DECL int pcre16_callout(pcre16_callout_block *);
#endif /* VPCOMPAT */

/* User defined callback which provides a stack just before the match starts. */

typedef pcre_jit_stack (*pcre_jit_callback)(void *);
typedef pcre16_jit_stack (*pcre16_jit_callback)(void *);

/* Exported PCRE functions */

PCRE_EXP_DECL pcre *pcre_compile(const char *, int, const char **, int *,
    const unsigned char *);
PCRE_EXP_DECL pcre16 *pcre16_compile(PCRE_SPTR16, int, const char **, int *,
    const unsigned char *);
PCRE_EXP_DECL pcre *pcre_compile2(const char *, int, int *, const char **,
    int *, const unsigned char *);
PCRE_EXP_DECL pcre16 *pcre16_compile2(PCRE_SPTR16, int, int *, const char **,
    int *, const unsigned char *);
PCRE_EXP_DECL int pcre_config(int, void *);
PCRE_EXP_DECL int pcre16_config(int, void *);
PCRE_EXP_DECL int pcre_copy_named_substring(const pcre *, const char *,
    int *, int, const char *, char *, int);
PCRE_EXP_DECL int pcre16_copy_named_substring(const pcre16 *, PCRE_SPTR16,
    int *, int, PCRE_SPTR16, PCRE_UCHAR16 *, int);
PCRE_EXP_DECL int pcre_copy_substring(const char *, int *, int, int,
    char *, int);
PCRE_EXP_DECL int pcre16_copy_substring(PCRE_SPTR16, int *, int, int,
    PCRE_UCHAR16 *, int);
PCRE_EXP_DECL int pcre_dfa_exec(const pcre *, const pcre_extra *,
    const char *, int, int, int, int *, int *, int);
PCRE_EXP_DECL int pcre16_dfa_exec(const pcre16 *, const pcre16_extra *,
    PCRE_SPTR16, int, int, int, int *, int *, int);
PCRE_EXP_DECL int pcre_exec(const pcre *, const pcre_extra *, PCRE_SPTR,
    int, int, int, int *, int);
PCRE_EXP_DECL int pcre16_exec(const pcre16 *, const pcre16_extra *,
    PCRE_SPTR16, int, int, int, int *, int);

```



```

PCRE_EXP_DECL void pcre_free_substring(const char *);
PCRE_EXP_DECL void pcre16_free_substring(PCRE_SPTR16);
PCRE_EXP_DECL void pcre_free_substring_list(const char **);
PCRE_EXP_DECL void pcre16_free_substring_list(PCRE_SPTR16 *);
PCRE_EXP_DECL int pcre_fullinfo(const pcre *, const pcre_extra *, int,
    void *);
PCRE_EXP_DECL int pcre16_fullinfo(const pcre16 *, const pcre16_extra *, int,
    void *);
PCRE_EXP_DECL int pcre_get_named_substring(const pcre *, const char *,
    int *, int, const char *, const char **);
PCRE_EXP_DECL int pcre16_get_named_substring(const pcre16 *, PCRE_SPTR16,
    int *, int, PCRE_SPTR16, PCRE_SPTR16 *);
PCRE_EXP_DECL int pcre_get_stringnumber(const pcre *, const char *);
PCRE_EXP_DECL int pcre16_get_stringnumber(const pcre16 *, PCRE_SPTR16);
PCRE_EXP_DECL int pcre_get_stringtable_entries(const pcre *, const char *,
    char **, char **);
PCRE_EXP_DECL int pcre16_get_stringtable_entries(const pcre16 *, PCRE_SPTR16,
    PCRE_UCHAR16 **, PCRE_UCHAR16 **);
PCRE_EXP_DECL int pcre_get_substring(const char *, int *, int, int,
    const char **);
PCRE_EXP_DECL int pcre16_get_substring(PCRE_SPTR16, int *, int, int,
    PCRE_SPTR16 *);
PCRE_EXP_DECL int pcre_get_substring_list(const char *, int *, int,
    const char ***);
PCRE_EXP_DECL int pcre16_get_substring_list(PCRE_SPTR16, int *, int,
    PCRE_SPTR16 **);
PCRE_EXP_DECL const unsigned char *pcre_maketables(void);
PCRE_EXP_DECL const unsigned char *pcre16_maketables(void);
PCRE_EXP_DECL int pcre_refcount(pcre *, int);
PCRE_EXP_DECL int pcre16_refcount(pcre16 *, int);
PCRE_EXP_DECL pcre_extra *pcre_study(const pcre *, int, const char **);
PCRE_EXP_DECL pcre16_extra *pcre16_study(const pcre16 *, int, const char **);
PCRE_EXP_DECL void pcre_free_study(pcre_extra *);
PCRE_EXP_DECL void pcre16_free_study(pcre16_extra *);
PCRE_EXP_DECL const char *pcre_version(void);
PCRE_EXP_DECL const char *pcre16_version(void);

/* Utility functions for byte order swaps. */
PCRE_EXP_DECL int pcre_pattern_to_host_byte_order(pcre *, pcre_extra *,
    const unsigned char *);
PCRE_EXP_DECL int pcre16_pattern_to_host_byte_order(pcre16 *, pcre16_extra *,
    const unsigned char *);
PCRE_EXP_DECL int pcre16_utf16_to_host_byte_order(PCRE_UCHAR16 *,
    PCRE_SPTR16, int, int *, int);

/* JIT compiler related functions. */

PCRE_EXP_DECL pcre_jit_stack *pcre_jit_stack_alloc(int, int);
PCRE_EXP_DECL pcre16_jit_stack *pcre16_jit_stack_alloc(int, int);
PCRE_EXP_DECL void pcre_jit_stack_free(pcre_jit_stack *);
PCRE_EXP_DECL void pcre16_jit_stack_free(pcre16_jit_stack *);
PCRE_EXP_DECL void pcre_assign_jit_stack(pcre_extra *,
    pcre_jit_callback, void *);
PCRE_EXP_DECL void pcre16_assign_jit_stack(pcre16_extra *,

```

```
        pcre16_jit_callback, void *);

#ifdef __cplusplus
} /* extern "C" */
#endif

#endif /* End of pcre.h */
```

1.41 glibc

1.41.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid

distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes

a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from

such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a

derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more

than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and

all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
 Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or
 modify it under the terms of the GNU Lesser General Public
 License as published by the Free Software Foundation; either
 version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
 License along with this library; if not, write to the Free Software
 Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
 school, if any, to sign a "copyright disclaimer" for the library, if
 necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
 library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
 Ty Coon, President of Vice

That's all there is to it!
 Copyright 1992, 1993, 1994, 1997 Henry Spencer. All rights reserved.
 This software is not subject to any license of the American Telephone
 and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on
 any computer system, and to alter it and redistribute it, subject
 to the following restrictions:

1. The author is not responsible for the consequences of use of this
 software, no matter how awful, even if they arise from flaws in it.
2. The origin of this software must not be misrepresented, either by
 explicit claim or by omission. Since few users ever read sources,
 credits must appear in the documentation.
3. Altered versions must be plainly marked as such, and must not be
 misrepresented as being the original software. Since few users
 ever read sources, credits must appear in the documentation.
4. This notice may not be removed or altered.

This file contains the copying permission notices for various files in the
 GNU C Library distribution that have copyright owners other than the Free
 Software Foundation. These notices all require that a copy of the notice
 be included in the accompanying documentation and be distributed with

binary distributions of the code, so be sure to include this file along with any binary distributions derived from the GNU C Library.

All code incorporated from 4.4 BSD is distributed under the following license:

Copyright (C) 1991 Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. [This condition was removed.]
4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The DNS resolver code, taken from BIND 4.9.5, is copyrighted by UC Berkeley, by Digital Equipment Corporation and by Internet Software Consortium. The DEC portions are under the following license:

Portions Copyright (C) 1993 by Digital Equipment Corporation.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies, and that the name of Digital Equipment Corporation not be used in advertising or publicity pertaining to distribution of the document or software without specific, written prior permission.

THE SOFTWARE IS PROVIDED ``AS IS" AND DIGITAL EQUIPMENT CORP. DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL DIGITAL EQUIPMENT CORPORATION BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING

FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The ISC portions are under the following license:

Portions Copyright (c) 1996-1999 by Internet Software Consortium.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The Sun RPC support (from rpcsrc-4.0) is covered by the following license:

Copyright (c) 2010, Oracle America, Inc.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the "Oracle America, Inc." nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following CMU license covers some of the support code for Mach,

derived from Mach 3.0:

Mach Operating System
Copyright (C) 1991,1990,1989 Carnegie Mellon University
All Rights Reserved.

Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof, and that both notices appear in supporting documentation.

CARNEGIE MELLON ALLOWS FREE USE OF THIS SOFTWARE IN ITS ``AS IS" CONDITION. CARNEGIE MELLON DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.

Carnegie Mellon requests users of this software to return to

Software Distribution Coordinator
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213-3890

or Software.Distribution@CS.CMU.EDU any improvements or extensions that they make and grant Carnegie Mellon the rights to redistribute these changes.

The file if_ppp.h is under the following CMU license:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY CARNEGIE MELLON UNIVERSITY AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE UNIVERSITY OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following license covers the files from Intel's "Highly Optimized Mathematical Functions for Itanium" collection:

Intel License Agreement

Copyright (c) 2000, Intel Corporation

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* The name of Intel Corporation may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL INTEL OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The files inet/getnameinfo.c and sysdeps/posix/getaddrinfo.c are copyright (C) by Craig Metz and are distributed under the following license:

/* The Inner Net License, Version 2.00

The author(s) grant permission for redistribution and use in source and binary forms, with or without modification, of the software and documentation provided that the following conditions are met:

0. If you receive a version of the software that is specifically labelled as not being for redistribution (check the version message and/or README), you are not permitted to redistribute that version of the software in any way or form.
1. All terms of the all other applicable copyrights and licenses must be followed.
2. Redistributions of source code must retain the authors' copyright notice(s), this list of conditions, and the following disclaimer.
3. Redistributions in binary form must reproduce the authors' copyright notice(s), this list of conditions, and the following disclaimer in the

- documentation and/or other materials provided with the distribution.
4. [The copyright holder has authorized the removal of this clause.]
 5. Neither the name(s) of the author(s) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ITS AUTHORS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

If these license terms cause you a real problem, contact the author. */

The file sunrpc/des_impl.c is copyright Eric Young:

Copyright (C) 1992 Eric Young
Collected from libdes and modified for SECURE RPC by Martin Kuck 1994
This file is distributed under the terms of the GNU Lesser General
Public License, version 2.1 or later - see the file COPYING.LIB for details.
If you did not receive a copy of the license with this program, please
see <<http://www.gnu.org/licenses/>> to obtain a copy.

The libidn code is copyright Simon Josefsson, with portions copyright
The Internet Society, Tom Tromey and Red Hat, Inc.:

Copyright (C) 2002, 2003, 2004, 2011 Simon Josefsson

This file is part of GNU Libidn.

GNU Libidn is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

GNU Libidn is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with GNU Libidn; if not, see <<http://www.gnu.org/licenses/>>.

The following notice applies to portions of libidn/nfkc.c:

This file contains functions from GLIB, including gutf8.c and
gunidecomp.c, all licensed under LGPL and copyright hold by:

Copyright (C) 1999, 2000 Tom Tromey

Copyright 2000 Red Hat, Inc.

The following applies to portions of libidn/punycode.c and libidn/punycode.h:

This file is derived from RFC 3492bis written by Adam M. Costello.

Disclaimer and license: Regarding this entire document or any portion of it (including the pseudocode and C code), the author makes no guarantees and is not responsible for any damage resulting from its use. The author grants irrevocable permission to anyone to use, modify, and distribute it in any way that does not diminish the rights of anyone else to use, modify, and distribute it, provided that redistributed derivative works do not contain misleading author or version information. Derivative works need not be licensed under similar terms.

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

The file inet/rcmd.c is under a UCB copyright and the following:

Copyright (C) 1998 WIDE Project.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the

- documentation and/or other materials provided with the distribution.
3. Neither the name of the project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE PROJECT OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The file `posix/runtests.c` is copyright Tom Lord:

Copyright 1995 by Tom Lord

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the copyright holder not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

Tom Lord DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL TOM LORD BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The `posix/rxspencer` tests are copyright Henry Spencer:

Copyright 1992, 1993, 1994, 1997 Henry Spencer. All rights reserved.
This software is not subject to any license of the American Telephone and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on any computer system, and to alter it and redistribute it, subject to the following restrictions:

1. The author is not responsible for the consequences of use of this software, no matter how awful, even if they arise from flaws in it.
2. The origin of this software must not be misrepresented, either by explicit claim or by omission. Since few users ever read sources,

credits must appear in the documentation.

3. Altered versions must be plainly marked as such, and must not be misrepresented as being the original software. Since few users ever read sources, credits must appear in the documentation.
4. This notice may not be removed or altered.

The file `posix/PCRE.tests` is copyright University of Cambridge:

Copyright (c) 1997-2003 University of Cambridge

Permission is granted to anyone to use this software for any purpose on any computer system, and to redistribute it freely, subject to the following restrictions:

1. This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
2. The origin of this software must not be misrepresented, either by explicit claim or by omission. In practice, this means that if you use PCRE in software that you distribute to others, commercially or otherwise, you must put a sentence like this

Regular expression support is provided by the PCRE library package, which is open source software, written by Philip Hazel, and copyright by the University of Cambridge, England.

somewhere reasonably visible in your documentation and in any relevant files or online help data or similar. A reference to the ftp site for the source, that is, to

`ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/`

should also be given in the documentation. However, this condition is not intended to apply to whole chains of software. If package A includes PCRE, it must acknowledge it, but if package B is software that includes package A, the condition is not imposed on package B (unless it uses PCRE independently).

3. Altered versions must be plainly marked as such, and must not be misrepresented as being the original software.
4. If PCRE is embedded in any software that is released under the GNU General Purpose Licence (GPL), or Lesser General Purpose Licence (LGPL), then the terms of that licence shall supersede any condition above with which it is incompatible.

Files from `Sun fdlbm` are copyright Sun Microsystems, Inc.:

Copyright (C) 1993 by Sun Microsystems, Inc. All rights reserved.

Developed at SunPro, a Sun Microsystems, Inc. business.
Permission to use, copy, modify, and distribute this

software is freely granted, provided that this notice is preserved.

Part of stdio-common/tst-printf.c is copyright C E Chew:

(C) Copyright C E Chew

Feel free to copy, use and distribute this software provided:

1. you do not pretend that you wrote it
2. you leave this copyright notice intact.

Various long double libm functions are copyright Stephen L. Moshier:

Copyright 2001 by Stephen L. Moshier <moshier@na-net.ornl.gov>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, see <<http://www.gnu.org/licenses/>>. */

1.42 gmp

1.42.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

GNU LESSER GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. [<http://fsf.org/>](http://fsf.org/)
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library

with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the

portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities,

conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate

copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of

Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the

original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions

either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Lesser General
Public License instead of this License.

1.43 gnutls

1.43.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that

you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General

Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means

all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library,

and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library

creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system,

rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus

excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting

redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

LICENSING

=====

Since GnuTLS version 3.1.10, the core library is released under the GNU Lesser General Public License (LGPL) version 2.1 or later (see doc/COPYING.LESSER for the license terms).

The GNU LGPL applies to the main GnuTLS library, while the included applications as well as gnutls-openssl library are under the GNU GPL version 3. The gnutls library is located in the lib/ and libdane/ directories, while the applications in src/ and, the gnutls-openssl library is at extra/.

The documentation in doc/ is under the GNU FDL license 1.3.

Note, however, that the nettle and the gmp libraries which are GnuTLS dependencies, they are distributed under a LGPLv3+ or GPLv2+ dual license. As such binaries linking to them need to adhere to either LGPLv3+ or the GPLv2+ license.

For any copyright year range specified as YYYY-ZZZZ in this package note that the range specifies every single year in that closed interval.

1.44 gperftools

1.44.1 Available under license

Copyright (c) 2005, Google Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.45 grep

1.45.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as

part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may

otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.46 gzip

1.46.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).
/* gzip.h -- common declarations for all gzip modules

Copyright (C) 1997-1999, 2001, 2006-2007, 2009-2016 Free Software Foundation, Inc.

Copyright (C) 1992-1993 Jean-loup Gailly.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. */

```
#ifdef __STDC__
typedef void *voidp;
#else
typedef char *voidp;
#endif
```

```

#ifndef __attribute__
# if __GNUC__ < 2 || (__GNUC__ == 2 && __GNUC_MINOR__ < 8) || __STRICT_ANSI__
#  define __attribute__(x)
# endif
#endif

#ifndef ATTRIBUTE_NORETURN
# define ATTRIBUTE_NORETURN __attribute__((__noreturn__))
#endif

/* I don't like nested includes, but the following headers are used
 * too often
 */
#include <stdio.h>
#include <sys/types.h> /* for off_t */
#include <time.h>
#include <string.h>
#define memzero(s, n) memset((voidp)(s), 0, (n))

#ifndef RETSIGTYPE
# define RETSIGTYPE void
#endif

#define local static

typedef unsigned char uch;
typedef unsigned short ush;
typedef unsigned long ulg;

/* Return codes from gzip */
#define OK 0
#define ERROR 1
#define WARNING 2

/* Compression methods (see algorithm.doc) */
#define STORED 0
#define COMPRESSED 1
#define PACKED 2
#define LZHED 3
/* methods 4 to 7 reserved */
#define DEFLATED 8
#define MAX_METHODS 9
extern int method; /* compression method */

/* To save memory for 16 bit systems, some arrays are overlaid between
 * the various modules:
 * deflate: prev+head window d_buf l_buf outbuf
 * unlwz: tab_prefix tab_suffix stack inbuf outbuf
 * inflate: window inbuf
 * ununpack: window inbuf prefix_len
 * unlzh: left+right window c_table inbuf c_len
 * For compression, input is done in window[]. For decompression, output
 * is done in window except for unlwz.
 */

```

```

#ifndef INBUFSIZ
# ifdef SMALL_MEM
#  define INBUFSIZ 0x2000 /* input buffer size */
# else
#  define INBUFSIZ 0x8000 /* input buffer size */
# endif
#endif
#define INBUF_EXTRA 64 /* required by unlwz() */

#ifndef OUTBUFSIZ
# ifdef SMALL_MEM
#  define OUTBUFSIZ 8192 /* output buffer size */
# else
#  define OUTBUFSIZ 16384 /* output buffer size */
# endif
#endif
#define OUTBUF_EXTRA 2048 /* required by unlwz() */

#ifndef DIST_BUFSIZE
# ifdef SMALL_MEM
#  define DIST_BUFSIZE 0x2000 /* buffer for distances, see trees.c */
# else
#  define DIST_BUFSIZE 0x8000 /* buffer for distances, see trees.c */
# endif
#endif

#ifndef DYN_ALLOC
# define EXTERN(type, array) extern type * near array
# define DECLARE(type, array, size) type * near array
# define ALLOC(type, array, size) { \
    array = (type*)fcalloc((size_t)((size)+1L)/2, 2*sizeof(type)); \
    if (!array) xalloc_die (); \
}
# define FREE(array) {if (array != NULL) fcfree(array), array=NULL;}
#else
# define EXTERN(type, array) extern type array[]
# define DECLARE(type, array, size) type array[size]
# define ALLOC(type, array, size)
# define FREE(array)
#endif

EXTERN(uch, inbuf); /* input buffer */
EXTERN(uch, outbuf); /* output buffer */
EXTERN(ush, d_buf); /* buffer for distances, see trees.c */
EXTERN(uch, window); /* Sliding window and suffix table (unlwz) */
#define tab_suffix window
#ifndef MAXSEG_64K
# define tab_prefix prev /* hash link (see deflate.c) */
# define head (prev+WSIZE) /* hash head (see deflate.c) */
    EXTERN(ush, tab_prefix); /* prefix code (see unlwz.c) */
#else
# define tab_prefix0 prev
# define head tab_prefix1
    EXTERN(ush, tab_prefix0); /* prefix for even codes */

```

```

    EXTERN(ush, tab_prefix1); /* prefix for odd codes */
#endif

extern unsigned insize; /* valid bytes in inbuf */
extern unsigned inptr; /* index of next byte to be processed in inbuf */
extern unsigned outcnt; /* bytes in output buffer */
extern int rsync; /* deflate into rsyncable chunks */

extern off_t bytes_in; /* number of input bytes */
extern off_t bytes_out; /* number of output bytes */
extern off_t header_bytes; /* number of bytes in gzip header */

extern int ifd; /* input file descriptor */
extern int ofd; /* output file descriptor */
extern char ifname[]; /* input file name or "stdin" */
extern char ofname[]; /* output file name or "stdout" */
extern char *program_name; /* program name */

extern struct timespec time_stamp; /* original time stamp (modification time) */
extern off_t ifile_size; /* input file size, -1 for devices (debug only) */

typedef int file_t; /* Do not use stdio */
#define NO_FILE (-1) /* in memory compression */

#define PACK_MAGIC "\037\036" /* Magic header for packed files */
#define GZIP_MAGIC "\037\213" /* Magic header for gzip files, 1F 8B */
#define OLD_GZIP_MAGIC "\037\236" /* Magic header for gzip 0.5 = freeze 1.x */
#define LZH_MAGIC "\037\240" /* Magic header for SCO LZH Compress files */
#define PKZIP_MAGIC "\120\113\003\004" /* Magic header for pkzip files */

/* gzip flag byte */
#define ASCII_FLAG 0x01 /* bit 0 set: file probably ascii text */
#define HEADER_CRC 0x02 /* bit 1 set: CRC16 for the gzip header */
#define EXTRA_FIELD 0x04 /* bit 2 set: extra field present */
#define ORIG_NAME 0x08 /* bit 3 set: original file name present */
#define COMMENT 0x10 /* bit 4 set: file comment present */
#define ENCRYPTED 0x20 /* bit 5 set: file is encrypted */
#define RESERVED 0xC0 /* bit 6,7: reserved */

/* internal file attribute */
#define UNKNOWN 0xffff
#define BINARY 0
#define ASCII 1

#ifndef WSIZE
# define WSIZE 0x8000 /* window size--must be a power of two, and */
#endif /* at least 32K for zip's deflate method */

#define MIN_MATCH 3
#define MAX_MATCH 258
/* The minimum and maximum match lengths */

#define MIN_LOOKAHEAD (MAX_MATCH+MIN_MATCH+1)
/* Minimum amount of lookahead, except at the end of the input file.

```

```

* See deflate.c for comments about the MIN_MATCH+1.
*/

#define MAX_DIST (WSIZE-MIN_LOOKAHEAD)
/* In order to simplify the code, particularly on 16 bit machines, match
* distances are limited to MAX_DIST instead of WSIZE.
*/

extern int exit_code; /* program exit code */
extern int verbose; /* be verbose (-v) */
extern int quiet; /* be quiet (-q) */
extern int level; /* compression level */
extern int test; /* check .z file integrity */
extern int to_stdout; /* output to stdout (-c) */
extern int save_orig_name; /* set if original name must be saved */

#define get_byte() (inptr < insize ? inbuf[inptr++] : fill_inbuf(0))
#define try_byte() (inptr < insize ? inbuf[inptr++] : fill_inbuf(1))

/* put_byte is used for the compressed output, put_ubyte for the
* uncompressed output. However unlzw() uses window for its
* suffix table instead of its output buffer, so it does not use put_ubyte
* (to be cleaned up).
*/
#define put_byte(c) {outbuf[outcnt++]=(uch)(c); if (outcnt==OUTBUFSIZ)\
flush_outbuf();}
#define put_ubyte(c) {window[outcnt++]=(uch)(c); if (outcnt==WSIZE)\
flush_window();}

/* Output a 16 bit value, lsb first */
#define put_short(w) \
{ if (outcnt < OUTBUFSIZ-2) { \
outbuf[outcnt++] = (uch)((w) & 0xff); \
outbuf[outcnt++] = (uch)((ush)(w) >> 8); \
} else { \
put_byte((uch)((w) & 0xff)); \
put_byte((uch)((ush)(w) >> 8)); \
} \
}

/* Output a 32 bit value to the bit stream, lsb first */
#define put_long(n) { \
put_short((n) & 0xffff); \
put_short(((ulg)(n)) >> 16); \
}

#define seekable() 0 /* force sequential output */
#define translate_eol 0 /* no option -a yet */

#define tolow(c) (isupper(c) ? tolower(c) : (c)) /* force to lower case */

/* Macros for getting two-byte and four-byte header values */
#define SH(p) ((ush)(uch)(p)[0] | ((ush)(uch)(p)[1] << 8))
#define LG(p) ((ulg)(SH(p)) | ((ulg)(SH(p)+2) << 16))

```

```

/* Diagnostic functions */
#ifdef DEBUG
# define Assert(cond,msg) {if (!(cond)) gzip_error (msg);}
# define Trace(x) fprintf x
# define Tracev(x) {if (verbose) fprintf x ;}
# define Tracevv(x) {if (verbose>1) fprintf x ;}
# define Tracec(c,x) {if (verbose && (c)) fprintf x ;}
# define Tracecv(c,x) {if (verbose>1 && (c)) fprintf x ;}
#else
# define Assert(cond,msg)
# define Trace(x)
# define Tracev(x)
# define Tracevv(x)
# define Tracec(c,x)
# define Tracecv(c,x)
#endif

#define WARN(msg) {if (!quiet) fprintf msg ; \
    if (exit_code == OK) exit_code = WARNING;}

/* in zip.c: */
extern int zip      (int in, int out);
extern int file_read (char *buf, unsigned size);

/* in unzip.c */
extern int unzip      (int in, int out);
extern int check_zipfile (int in);

/* in unpack.c */
extern int unpack      (int in, int out);

/* in unlzh.c */
extern int unlzh      (int in, int out);

/* in gzip.c */
extern void abort_gzip (void) ATTRIBUTE_NORETURN;

/* in deflate.c */
extern void lm_init (int pack_level, ush *flags);
extern off_t deflate (void);

/* in trees.c */
extern void ct_init      (ush *attr, int *method);
extern int ct_tally      (int dist, int lc);
extern off_t flush_block (char *buf, ulg stored_len, int pad, int eof);

/* in bits.c */
extern void bi_init      (file_t zipfile);
extern void send_bits      (int value, int length);
extern unsigned bi_reverse (unsigned value, int length) _GL_ATTRIBUTE_CONST;
extern void bi_windup      (void);
extern void copy_block      (char *buf, unsigned len, int header);
extern int (*read_buf)      (char *buf, unsigned size);

```



```
/* in util.c: */
extern int copy      (int in, int out);
extern ulg updcrc    (uch *s, unsigned n);
extern void clear_bufs (void);
extern int fill_inbuf (int eof_ok);
extern void flush_outbuf (void);
extern void flush_window (void);
extern void write_buf (int fd, voidp buf, unsigned cnt);
extern int read_buffer (int fd, voidp buf, unsigned int cnt);
extern char *strlwr   (char *s);
extern char *gzip_base_name (char *fname) _GL_ATTRIBUTE_PURE;
extern int xunlink     (char *fname);
extern void make_simple_name (char *name);
extern char *add_envopt (int *argcp, char ***argvp, char const *env);
extern void gzip_error  (char const *m) ATTRIBUTE_NORETURN;
extern void xalloc_die  (void) ATTRIBUTE_NORETURN;
extern void warning     (char const *m);
extern void read_error  (void) ATTRIBUTE_NORETURN;
extern void write_error (void) ATTRIBUTE_NORETURN;
extern void display_ratio (off_t num, off_t den, FILE *file);
extern void fprintf_off  (FILE *, off_t, int);

/* in inflate.c */
extern int inflate (void);
```

1.47 hwdata

1.47.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

This data is licenced under 2 different licenses

- 1) GNU General Public License, version 2 or later
- 2) XFree86 1.0 license

This data can be used freely under either license.

1.48 i2c-tools

1.48.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.49 ifupdown

1.49.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.50 inetutils

1.50.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.51 init-ifupdown

1.51.1 Available under license

This package was created by Peter Tobias tobias@et-inf.fho-emden.de on Wed, 24 Aug 1994 21:33:28 +0200 and maintained by Anthony Towns <ajt@debian.org> until 2001.
It is currently maintained by Marco d'Itri <md@linux.it>.

Copyright 1994-2010 Peter Tobias, Anthony Towns and Marco d'Itri

The programs in this package are distributed under the terms of the GNU General Public License, version 2 as distributed by the Free Software Foundation. On Debian systems, a copy of this license may be found in /usr/share/common-licenses/GPL-2.

1.52 initscripts

1.52.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.53 iproute2

1.53.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

```
/*
 * ip.c"ip" utility frontend.
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version
 * 2 of the License, or (at your option) any later version.
 *
 * Authors: Alexey Kuznetsov, <kuznet@ms2.inr.ac.ru>
 */
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <syslog.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <errno.h>

#include "SNAPSHOT.h"
#include "utils.h"
#include "ip_common.h"
#include "namespace.h"
#include "color.h"
```

```
int preferred_family = AF_UNSPEC;
int human_readable;
int use_iec;
int show_stats;
int show_details;
int resolve_hosts;
int online;
int brief;
```

```

int timestamp;
const char *_SL_;
int force;
int max_flush_loops = 10;
int batch_mode;
bool do_all;

struct rtnl_handle rth = { .fd = -1 };

static void usage(void) __attribute__((noreturn));

static void usage(void)
{
    fprintf(stderr,
"Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }\n"
"       ip [ -force ] -batch filename\n"
"where OBJECT := { link | address | addrlabel | route | rule | neigh | ntable |\n"
"               tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm |\n"
"               netns | l2tp | fou | macsec | tcp_metrics | token | netconf }\n"
"OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |\n"
"            -h[uman-readable] | -iec |\n"
"            -f[amily] { inet | inet6 | ipx | dnet | mpls | bridge | link } |\n"
"            -4 | -6 | -I | -D | -B | -O |\n"
"            -l[oops] { maximum-addr-flush-attempts } | -br[ief] |\n"
"            -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename] |\n"
"            -rc[vbuf] [size] | -n[etns] name | -a[ll] | -c[olor] }\n");
    exit(-1);
}

static int do_help(int argc, char **argv)
{
    usage();
    return 0;
}

static const struct cmd {
    const char *cmd;
    int (*func)(int argc, char **argv);
} cmds[] = {
    { "address", do_ipaddr },
    { "addrlabel", do_ipaddrlabel },
    { "maddress", do_multiaddr },
    { "route", do_iproute },
    { "rule", do_iprule },
    { "neighbor", do_ipneigh },
    { "neighbour", do_ipneigh },
    { "ntable", do_ipntable },
    { "ntbl", do_ipntable },
    { "link", do_iplink },
    { "l2tp", do_ipl2tp },
    { "fou", do_ipfou },
    { "macsec", do_ipmacsec },
    { "tunnel", do iptunnel },
    { "tunl", do iptunnel },

```

```

{ "tuntap",do_ip tuntap },
{ "tap",do_ip tuntap },
{ "token",do_ip token },
{ "tcpmetrics",do_tcp_metrics },
{ "tcp_metrics", do_tcp_metrics },
{ "monitor",do_ipmonitor },
{ "xfrm",do_xfrm },
{ "mroute",do_multiroute },
{ "mrule",do_multirule },
{ "netns",do_netns },
{ "netconf",do_ipnetconf },
{ "help",do_help },
{ 0 }
};

static int do_cmd(const char *argv0, int argc, char **argv)
{
    const struct cmd *c;

    for (c = cmds; c->cmd; ++c) {
        if (matches(argv0, c->cmd) == 0)
            return -(c->func(argc-1, argv+1));
    }

    fprintf(stderr, "Object \"%s\" is unknown, try \"ip help\".\n", argv0);
    return EXIT_FAILURE;
}

static int batch(const char *name)
{
    char *line = NULL;
    size_t len = 0;
    int ret = EXIT_SUCCESS;
    int orig_family = preferred_family;

    batch_mode = 1;

    if (name && strcmp(name, "-") != 0) {
        if (freopen(name, "r", stdin) == NULL) {
            fprintf(stderr,
                "Cannot open file \"%s\" for reading: %s\n",
                name, strerror(errno));
            return EXIT_FAILURE;
        }
    }

    if (rtnl_open(&rth, 0) < 0) {
        fprintf(stderr, "Cannot open rtnetlink\n");
        return EXIT_FAILURE;
    }

    cmdlineno = 0;
    while (getcmdline(&line, &len, stdin) != -1) {
        char *largv[100];
        int largc;

```

```
preferred_family = orig_family;

largc = makeargs(line, largv, 100);
if (largc == 0)
continue; /* blank line */

if (do_cmd(largv[0], largc, largv)) {
fprintf(stderr, "Command failed %s:%d\n",
name, cmdlineno);
ret = EXIT_FAILURE;
if (!force)
break;
}
}
if (line)
free(line);

rtnl_close(&rth);
return ret;
}

int main(int argc, char **argv)
{
char *basename;
char *batch_file = NULL;

basename = strrchr(argv[0], '/');
if (basename == NULL)
basename = argv[0];
else
basename++;

while (argc > 1) {
char *opt = argv[1];

if (strcmp(opt, "--") == 0) {
argc--; argv++;
break;
}
if (opt[0] != '-')
break;
if (opt[1] == '-')
opt++;
if (matches(opt, "-loops") == 0) {
argc--;
argv++;
if (argc <= 1)
usage();
max_flush_loops = atoi(argv[1]);
} else if (matches(opt, "-family") == 0) {
argc--;
argv++;
}
```

```

if (argc <= 1)
usage();
if (strcmp(argv[1], "help") == 0)
usage();
else
preferred_family = read_family(argv[1]);
if (preferred_family == AF_UNSPEC)
invarg("invalid protocol family", argv[1]);
} else if (strcmp(opt, "-4") == 0) {
preferred_family = AF_INET;
} else if (strcmp(opt, "-6") == 0) {
preferred_family = AF_INET6;
} else if (strcmp(opt, "-0") == 0) {
preferred_family = AF_PACKET;
} else if (strcmp(opt, "-I") == 0) {
preferred_family = AF_IPX;
} else if (strcmp(opt, "-D") == 0) {
preferred_family = AF_DECnet;
} else if (strcmp(opt, "-M") == 0) {
preferred_family = AF_MPLS;
} else if (strcmp(opt, "-B") == 0) {
preferred_family = AF_BRIDGE;
} else if (matches(opt, "-human") == 0 ||
matches(opt, "-human-readable") == 0) {
++human_readable;
} else if (matches(opt, "-iec") == 0) {
++use_iec;
} else if (matches(opt, "-stats") == 0 ||
matches(opt, "-statistics") == 0) {
++show_stats;
} else if (matches(opt, "-details") == 0) {
++show_details;
} else if (matches(opt, "-resolve") == 0) {
++resolve_hosts;
} else if (matches(opt, "-online") == 0) {
++online;
} else if (matches(opt, "-timestamp") == 0) {
++timestamp;
} else if (matches(opt, "-tshort") == 0) {
++timestamp;
++timestamp_short;
}
} else if (matches(opt, "-numeric") == 0) {
rtnl_names_numeric++;
}
} else if (matches(opt, "-Version") == 0) {
printf("ip utility, iproute2-ss%s\n", SNAPSHOT);
exit(0);
} else if (matches(opt, "-force") == 0) {
++force;
} else if (matches(opt, "-batch") == 0) {
argc--;
argv++;
}
if (argc <= 1)
usage();

```

```

batch_file = argv[1];
} else if (matches(opt, "-brief") == 0) {
++brief;
} else if (matches(opt, "-rcvbuf") == 0) {
unsigned int size;

argc--;
argv++;
if (argc <= 1)
usage();
if (get_unsigned(&size, argv[1], 0)) {
fprintf(stderr, "Invalid rcvbuf size \"%s\"\n",
argv[1]);
exit(-1);
}
rcvbuf = size;
} else if (matches(opt, "-color") == 0) {
enable_color();
} else if (matches(opt, "-help") == 0) {
usage();
} else if (matches(opt, "-netns") == 0) {
NEXT_ARG();
if (netns_switch(argv[1]))
exit(-1);
} else if (matches(opt, "-all") == 0) {
do_all = true;
} else {
fprintf(stderr,
"Option \"%s\" is unknown, try \"ip -help\".\n",
opt);
exit(-1);
}
argc--;argv++;
}

_SL_ = oneline ? "\\\" : "\n";

if (batch_file)
return batch(batch_file);

if (rtnl_open(&rth, 0) < 0)
exit(1);

if (strlen(basename) > 2)
return do_cmd(basename+2, argc, argv);

if (argc > 1)
return do_cmd(argv[1], argc-1, argv+1);

rtnl_close(&rth);
usage();
}

```

1.54 iptables

1.54.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

/* Code to take an iptables-style command line and do it. */

/*

* Author: Paul.Russell@rustcorp.com.au and mneuling@radlogic.com.au

*

* (C) 2000-2002 by the netfilter coreteam <coreteam@netfilter.org>:

* Paul 'Rusty' Russell <rusty@rustcorp.com.au>

* Marc Boucher <marc+nf@mbsi.ca>

* James Morris <jmorris@intercode.com.au>

* Harald Welte <laforge@gnumonks.org>

* Jozsef Kadlecsek <kadlec@blackhole.kfki.hu>

*

*This program is free software; you can redistribute it and/or modify

*it under the terms of the GNU General Public License as published by

*the Free Software Foundation; either version 2 of the License, or

*(at your option) any later version.

*

*This program is distributed in the hope that it will be useful,

*but WITHOUT ANY WARRANTY; without even the implied warranty of

*MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

*GNU General Public License for more details.

*

*You should have received a copy of the GNU General Public License

*along with this program; if not, write to the Free Software

*Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

*/

#include <getopt.h>

#include <string.h>

#include <netdb.h>

#include <errno.h>

#include <stdbool.h>

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <stdarg.h>

```

#include <limits.h>
#include <unistd.h>
#include <iptables.h>
#include <xtables.h>
#include <fcntl.h>
#include "xshared.h"

#ifndef TRUE
#define TRUE 1
#endif
#ifndef FALSE
#define FALSE 0
#endif

#define CMD_NONE0x0000U
#define CMD_INSERT0x0001U
#define CMD_DELETE0x0002U
#define CMD_DELETE_NUM0x0004U
#define CMD_REPLACE0x0008U
#define CMD_APPEND0x0010U
#define CMD_LIST0x0020U
#define CMD_FLUSH0x0040U
#define CMD_ZERO0x0080U
#define CMD_NEW_CHAIN0x0100U
#define CMD_DELETE_CHAIN0x0200U
#define CMD_SET_POLICY0x0400U
#define CMD_RENAME_CHAIN0x0800U
#define CMD_LIST_RULES0x1000U
#define CMD_ZERO_NUM0x2000U
#define CMD_CHECK0x4000U
#define NUMBER_OF_CMD16
static const char cmdflags[] = { 'T', 'D', 'D', 'R', 'A', 'L', 'F', 'Z',
    'N', 'X', 'P', 'E', 'S', 'Z', 'C' };

#define OPT_FRAGMENT 0x00800U
#define NUMBER_OF_OPTARRAY_SIZE(optflags)
static const char optflags[]
= { 'n', 's', 'd', 'p', 'j', 'v', 'x', 'i', 'o', '0', 'c', 'f' };

static struct option original_opts[] = {
    { .name = "append", .has_arg = 1, .val = 'A' },
    { .name = "delete", .has_arg = 1, .val = 'D' },
    { .name = "check", .has_arg = 1, .val = 'C' },
    { .name = "insert", .has_arg = 1, .val = 'T' },
    { .name = "replace", .has_arg = 1, .val = 'R' },
    { .name = "list", .has_arg = 2, .val = 'L' },
    { .name = "list-rules", .has_arg = 2, .val = 'S' },
    { .name = "flush", .has_arg = 2, .val = 'F' },
    { .name = "zero", .has_arg = 2, .val = 'Z' },
    { .name = "new-chain", .has_arg = 1, .val = 'N' },
    { .name = "delete-chain", .has_arg = 2, .val = 'X' },
    { .name = "rename-chain", .has_arg = 1, .val = 'E' },
    { .name = "policy", .has_arg = 1, .val = 'P' },
    { .name = "source", .has_arg = 1, .val = 's' },

```



```

/*FLUSH*/    {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*ZERO*/     {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*NEW_CHAIN*/ {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*DEL_CHAIN*/ {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*SET_POLICY*/ {'x','x','x','x','x',' ','x','x','x','x',' ','x'},
/*RENAME*/    {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*LIST_RULES*/ {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*ZERO_NUM*/  {'x','x','x','x','x',' ','x','x','x','x','x','x'},
/*CHECK*/     {'x',' ',' ',' ',' ',' ','x',' ',' ','x','x',' '},
};

```

```

static const int inverse_for_options[NUMBER_OF_OPT] =
{
/* -n */ 0,
/* -s */ IPT_INV_SRCIP,
/* -d */ IPT_INV_DSTIP,
/* -p */ XT_INV_PROTO,
/* -j */ 0,
/* -v */ 0,
/* -x */ 0,
/* -i */ IPT_INV_VIA_IN,
/* -o */ IPT_INV_VIA_OUT,
/* --line */ 0,
/* -c */ 0,
/* -f */ IPT_INV_FRAG,
};

```

```

#define opts iptables_globals.opts
#define prog_name iptables_globals.program_name
#define prog_vers iptables_globals.program_version

```

```

static void __attribute__((noreturn))
exit_tryhelp(int status)
{
if (line != -1)
fprintf(stderr, "Error occurred at line: %d\n", line);
fprintf(stderr, "Try '%s -h' or '%s --help' for more information.\n",
prog_name, prog_name);
xtables_free_opts(1);
exit(status);
}

```

```

static void
exit_printhelp(const struct xtables_rule_match *matches)
{
printf("%s v%s\n\n",
"Usage: %s -[ACD] chain rule-specification [options]\n"
"      %s -I chain [rulenum] rule-specification [options]\n"
"      %s -R chain rulenum rule-specification [options]\n"
"      %s -D chain rulenum [options]\n"
"      %s -[LS] [chain [rulenum]] [options]\n"
"      %s -[FZ] [chain] [options]\n"
"      %s -[NX] chain\n"
"      %s -E old-chain-name new-chain-name\n"

```

```

"    %s -P chain target [options]\n"
"    %s -h (print this help information)\n\n",
    prog_name, prog_vers, prog_name, prog_name,
    prog_name, prog_name, prog_name, prog_name,
    prog_name, prog_name, prog_name, prog_name);

printf(
"Commands:\n"
"Either long or short options are allowed.\n"
"--append -A chainAppend to chain\n"
"--check -C chainCheck for the existence of a rule\n"
"--delete -D chainDelete matching rule from chain\n"
"--delete -D chain rulenum\n"
"Delete rule rulenum (1 = first) from chain\n"
"--insert -I chain [rulenum]\n"
"Insert in chain as rulenum (default 1=first)\n"
"--replace -R chain rulenum\n"
"Replace rule rulenum (1 = first) in chain\n"
"--list -L [chain [rulenum]]\n"
"List the rules in a chain or all chains\n"
"--list-rules -S [chain [rulenum]]\n"
"Print the rules in a chain or all chains\n"
"--flush -F [chain]Delete all rules in chain or all chains\n"
"--zero -Z [chain [rulenum]]\n"
"Zero counters in chain or all chains\n"
"--new -N chainCreate a new user-defined chain\n"
"--delete-chain\n"
"-X [chain]Delete a user-defined chain\n"
"--policy -P chain target\n"
"Change policy on chain to target\n"
"--rename-chain\n"
"-E old-chain new-chain\n"
"Change chain name, (moving any references)\n"

"Options:\n"
"--ipv4-4Nothing (line is ignored by ip6tables-restore)\n"
"--ipv6-6Error (line is ignored by iptables-restore)\n"
"[!] --protocol-p protoprotocol: by number or name, eg. `tcp`\n"
"[!] --source-s address[/mask][...]\n"
"source specification\n"
"[!] --destination -d address[/mask][...]\n"
"destination specification\n"
"[!] --in-interface -i input name[+]\n"
"network interface name ([+] for wildcard)\n"
"--jump-j target\n"
"target for rule (may load target extension)\n"
#ifdef IPT_F_GOTO
"--goto -g chain\n"
"        jump to chain with no return\n"
#endif
"--match-m match\n"
"extended match (may load extension)\n"
"--numeric-numeric output of addresses and ports\n"
"[!] --out-interface -o output name[+]\n"
"network interface name ([+] for wildcard)\n"

```

```

" --table-t tabletable to manipulate (default: `filter')\n"
" --verbose-vverbose mode\n"
" --wait-w [seconds]wait for the xtables lock\n"
" --line-numbersprint line numbers when listing\n"
" --exact-xexpand numbers (display exact values)\n"
"[!] --fragment-fmatch second or further fragments only\n"
" --modprobe=<command>try to insert modules using this command\n"
" --set-counters PKTS BYTESset the counter during insert/append\n"
"[!] --version-Vprint package version.\n");

print_extension_helps(xtables_targets, matches);
exit(0);
}

void
iptables_exit_error(enum xtables_exittype status, const char *msg, ...)
{
    va_list args;

    va_start(args, msg);
    fprintf(stderr, "%s v%s: ", prog_name, prog_vers);
    vfprintf(stderr, msg, args);
    va_end(args);
    fprintf(stderr, "\n");
    if (status == PARAMETER_PROBLEM)
        exit_tryhelp(status);
    if (status == VERSION_PROBLEM)
        fprintf(stderr,
            "Perhaps iptables or your kernel needs to be upgraded.\n");
    /* On error paths, make sure that we don't leak memory */
    xtables_free_opts(1);
    exit(status);
}

static void
generic_opt_check(int command, int options)
{
    int i, j, legal = 0;

    /* Check that commands are valid with options. Complicated by the
     * fact that if an option is legal with *any* command given, it is
     * legal overall (ie. -z and -l).
     */
    for (i = 0; i < NUMBER_OF_OPT; i++) {
        legal = 0; /* -1 => illegal, 1 => legal, 0 => undecided. */

        for (j = 0; j < NUMBER_OF_CMD; j++) {
            if (!(command & (1<<j)))
                continue;

            if (!(options & (1<<i))) {
                if (commands_v_options[j][i] == '+')
                    xtables_error(PARAMETER_PROBLEM,
                        "You need to supply the `-%c' "

```

```

        "option for this command\n",
        optflags[i]);
    } else {
    if (commands_v_options[j][i] != 'x')
    legal = 1;
    else if (legal == 0)
    legal = -1;
    }
    }
    if (legal == -1)
    xtables_error(PARAMETER_PROBLEM,
        "Illegal option `-%c' with this command\n",
        optflags[i]);
    }
    }

static char
opt2char(int option)
{
    const char *ptr;
    for (ptr = optflags; option > 1; option >= 1, ptr++);

    return *ptr;
}

static char
cmd2char(int option)
{
    const char *ptr;
    for (ptr = cmdflags; option > 1; option >= 1, ptr++);

    return *ptr;
}

static void
add_command(unsigned int *cmd, const int newcmd, const int othercmds,
            int invert)
{
    {
    if (invert)
    xtables_error(PARAMETER_PROBLEM, "unexpected ! flag");
    if (*cmd & (~othercmds))
    xtables_error(PARAMETER_PROBLEM, "Cannot use -%c with -%c\n",
        cmd2char(newcmd), cmd2char(*cmd & (~othercmds)));
    *cmd |= newcmd;
    }
}

/*
 * All functions starting with "parse" should succeed, otherwise
 * the program fails.
 * Most routines return pointers to static data that may change
 * between calls to the same or other routines with a few exceptions:
 * "host_to_addr", "parse_hostnetwork", and "parse_hostnetworkmask"
 * return global static data.
 */

```

```
/* Christophe Burki wants -p 6' to imply -m tcp'. */
/* Can't be zero. */
static int
parse_rulenum(const char *rule)
{
    unsigned int rulenum;

    if (!xtables_strtoui(rule, NULL, &rulenum, 1, INT_MAX))
        xtables_error(PARAMETER_PROBLEM,
            "Invalid rule number '%s'", rule);

    return rulenum;
}

static void
parse_chain(const char *chainname)
{
    const char *ptr;

    if (strlen(chainname) >= XT_EXTENSION_MAXNAMELEN)
        xtables_error(PARAMETER_PROBLEM,
            "chain name '%s' too long (must be under %u chars)",
            chainname, XT_EXTENSION_MAXNAMELEN);

    if (*chainname == '-' || *chainname == '!')
        xtables_error(PARAMETER_PROBLEM,
            "chain name not allowed to start "
            "with '%c'\n", *chainname);

    if (xtables_find_target(chainname, XTF_TRY_LOAD))
        xtables_error(PARAMETER_PROBLEM,
            "chain name may not clash "
            "with target name\n");

    for (ptr = chainname; *ptr; ptr++)
        if (isspace(*ptr))
            xtables_error(PARAMETER_PROBLEM,
                "Invalid chain name '%s'", chainname);
}

static const char *
parse_target(const char *targetname)
{
    const char *ptr;

    if (strlen(targetname) < 1)
        xtables_error(PARAMETER_PROBLEM,
            "Invalid target name (too short)");

    if (strlen(targetname) >= XT_EXTENSION_MAXNAMELEN)
        xtables_error(PARAMETER_PROBLEM,
            "Invalid target name '%s' (%u chars max)",
            targetname, XT_EXTENSION_MAXNAMELEN - 1);
}
```

```

for (ptr = targetname; *ptr; ptr++)
if (isspace(*ptr))
xtables_error(PARAMETER_PROBLEM,
    "Invalid target name '%s'", targetname);
return targetname;
}

static void
set_option(unsigned int *options, unsigned int option, uint8_t *invflg,
    int invert)
{
if (*options & option)
xtables_error(PARAMETER_PROBLEM, "multiple -%c flags not allowed",
    opt2char(option));
*options |= option;

if (invert) {
unsigned int i;
for (i = 0; i << i != option; i++);

if (!inverse_for_options[i])
xtables_error(PARAMETER_PROBLEM,
    "cannot have ! before -%c",
    opt2char(option));
*invflg |= inverse_for_options[i];
}
}

static void
print_header(unsigned int format, const char *chain, struct xtc_handle *handle)
{
struct xt_counters counters;
const char *pol = iptc_get_policy(chain, &counters, handle);
printf("Chain %s", chain);
if (pol) {
printf(" (policy %s", pol);
if (!(format & FMT_NOCOUNTS)) {
fputc(' ', stdout);
xtables_print_num(counters.pcnt, (format|FMT_NOTABLE));
fputs("packets, ", stdout);
xtables_print_num(counters.bcnt, (format|FMT_NOTABLE));
fputs("bytes", stdout);
}
printf(")\n");
} else {
unsigned int refs;
if (!iptc_get_references(&refs, chain, handle))
printf(" (ERROR obtaining refs)\n");
else
printf(" (%u references)\n", refs);
}

if (format & FMT_LINENUMBERS)
printf(FMT("%-4s ", "%s "), "num");
if (!(format & FMT_NOCOUNTS)) {

```

```

    if (format & FMT_KILOMEGAGIGA) {
        printf(FMT("%5s ", "%s "), "pkts");
        printf(FMT("%5s ", "%s "), "bytes");
    } else {
        printf(FMT("%8s ", "%s "), "pkts");
        printf(FMT("%10s ", "%s "), "bytes");
    }
}

if (!(format & FMT_NOTARGET))
    printf(FMT("%-9s ", "%s "), "target");
fputs(" prot ", stdout);
if (format & FMT_OPTIONS)
    fputs("opt", stdout);
if (format & FMT_VIA) {
    printf(FMT(" %-6s ", "%s "), "in");
    printf(FMT("%-6s ", "%s "), "out");
}
printf(FMT(" %-19s ", "%s "), "source");
printf(FMT(" %-19s ", "%s "), "destination");
printf("\n");
}

static int
print_match(const struct xt_entry_match *m,
            const struct ipt_ip *ip,
            int numeric)
{
    const struct xtables_match *match =
        xtables_find_match(m->u.user.name, XTF_TRY_LOAD, NULL);

    if (match) {
        if (match->print)
            match->print(ip, m, numeric);
        else
            printf("%s ", match->name);
    } else {
        if (m->u.user.name[0])
            printf("UNKNOWN match '%s' ", m->u.user.name);
    }
    /* Don't stop iterating. */
    return 0;
}

/* e is called 'fw' here for historical reasons */
static void
print_firewall(const struct ipt_entry *fw,
               const char *targname,
               unsigned int num,
               unsigned int format,
               struct xtc_handle *const handle)
{
    const struct xtables_target *target = NULL;
    const struct xt_entry_target *t;

```

```

uint8_t flags;
char buf[BUFSIZ];

if (!iptc_is_chain(targname, handle))
target = xtables_find_target(targname, XTF_TRY_LOAD);
else
target = xtables_find_target(XT_STANDARD_TARGET,
                             XTF_LOAD_MUST_SUCCEED);

t = ipt_get_target((struct ipt_entry *)fw);
flags = fw->ip.flags;

if (format & FMT_LINENUMBERS)
printf(FMT("%-4u ", "%u "), num);

if (!(format & FMT_NOCOUNTS)) {
xtables_print_num(fw->counters.pcnt, format);
xtables_print_num(fw->counters.bcnc, format);
}

if (!(format & FMT_NOTARGET))
printf(FMT("%-9s ", "%s "), targname);

fputc(fw->ip.invflags & XT_INV_PROTO ? '!' : '-', stdout);
{
const char *pname = proto_to_name(fw->ip.proto, format&FMT_NUMERIC);
if (pname)
printf(FMT("%-5s", "%s "), pname);
else
printf(FMT("%-5hu", "%hu "), fw->ip.proto);
}

if (format & FMT_OPTIONS) {
if (format & FMT_NOTABLE)
fputs("opt ", stdout);
fputc(fw->ip.invflags & IPT_INV_FRAG ? '!' : '-', stdout);
fputc(flags & IPT_F_FRAG ? 'f' : '-', stdout);
fputc(' ', stdout);
}

if (format & FMT_VIA) {
char iface[IFNAMSIZ+2];

if (fw->ip.invflags & IPT_INV_VIA_IN) {
iface[0] = '!';
iface[1] = '\0';
}
else iface[0] = '\0';

if (fw->ip.iniface[0] != '\0') {
strcat(iface, fw->ip.iniface);
}
else if (format & FMT_NUMERIC) strcat(iface, "*");
else strcat(iface, "any");
printf(FMT(" %-6s ", "in %s "), iface);
}

```



```

if (fw->ip.invflags & IPT_INV_VIA_OUT) {
    iface[0] = '!';
    iface[1] = '\0';
}
else iface[0] = '\0';

if (fw->ip.outiface[0] != '\0') {
    strcat(iface, fw->ip.outiface);
}
else if (format & FMT_NUMERIC) strcat(iface, "*");
else strcat(iface, "any");
printf(FMT("%-6s ", "out %s "), iface);
}

fputc(fw->ip.invflags & IPT_INV_SRCIP ? '!' : ' ', stdout);
if (fw->ip.smask.s_addr == 0L && !(format & FMT_NUMERIC))
    printf(FMT("%-19s ", "%s "), "anywhere");
else {
    if (format & FMT_NUMERIC)
        strcpy(buf, xtables_ipaddr_to_numeric(&fw->ip.src));
    else
        strcpy(buf, xtables_ipaddr_to_anycname(&fw->ip.src));
    strcat(buf, xtables_ipmask_to_numeric(&fw->ip.smask));
    printf(FMT("%-19s ", "%s "), buf);
}

fputc(fw->ip.invflags & IPT_INV_DSTIP ? '!' : ' ', stdout);
if (fw->ip.dmask.s_addr == 0L && !(format & FMT_NUMERIC))
    printf(FMT("%-19s ", "-> %s"), "anywhere");
else {
    if (format & FMT_NUMERIC)
        strcpy(buf, xtables_ipaddr_to_numeric(&fw->ip.dst));
    else
        strcpy(buf, xtables_ipaddr_to_anycname(&fw->ip.dst));
    strcat(buf, xtables_ipmask_to_numeric(&fw->ip.dmask));
    printf(FMT("%-19s ", "-> %s"), buf);
}

if (format & FMT_NOTABLE)
    fputs(" ", stdout);

#ifdef IPT_F_GOTO
if (fw->ip.flags & IPT_F_GOTO)
    printf("[goto] ");
#endif

IPT_MATCH_ITERATE(fw, print_match, &fw->ip, format & FMT_NUMERIC);

if (target) {
    if (target->print)
        /* Print the target information. */
        target->print(&fw->ip, t, format & FMT_NUMERIC);
    } else if (t->u.target_size != sizeof(*t))

```

```

printf("[%u bytes of unknown target data] ",
      (unsigned int)(t->u.target_size - sizeof(*t)));

if (!(format & FMT_NONEWLINE))
fputc('\n', stdout);
}

static void
print_firewall_line(const struct ipt_entry *fw,
                    struct xtc_handle *const h)
{
struct xt_entry_target *t;

t = ipt_get_target((struct ipt_entry *)fw);
print_firewall(fw, t->u.user.name, 0, FMT_PRINT_RULE, h);
}

static int
append_entry(const xt_chainlabel chain,
             struct ipt_entry *fw,
             unsigned int nsaddrs,
             const struct in_addr saddrs[],
             const struct in_addr smasks[],
             unsigned int ndaddrs,
             const struct in_addr daddrs[],
             const struct in_addr dmask[],
             int verbose,
             struct xtc_handle *handle)
{
unsigned int i, j;
int ret = 1;

for (i = 0; i < nsaddrs; i++) {
fw->ip.src.s_addr = saddrs[i].s_addr;
fw->ip.smask.s_addr = smasks[i].s_addr;
for (j = 0; j < ndaddrs; j++) {
fw->ip.dst.s_addr = daddrs[j].s_addr;
fw->ip.dmask.s_addr = dmask[j].s_addr;
if (verbose)
print_firewall_line(fw, handle);
ret &= iptc_append_entry(chain, fw, handle);
}
}

return ret;
}

static int
replace_entry(const xt_chainlabel chain,
             struct ipt_entry *fw,
             unsigned int rulenum,
             const struct in_addr *saddr, const struct in_addr *smask,
             const struct in_addr *daddr, const struct in_addr *dmask,
             int verbose,
             struct xtc_handle *handle)

```

```

{
fw->ip.src.s_addr = saddr->s_addr;
fw->ip.dst.s_addr = daddr->s_addr;
fw->ip.smask.s_addr = smask->s_addr;
fw->ip.dmask.s_addr = dmask->s_addr;

if(verbose)
print_firewall_line(fw, handle);
return iptc_replace_entry(chain, fw, rulenum, handle);
}

static int
insert_entry(const xt_chainlabel chain,
    struct ipt_entry *fw,
    unsigned int rulenum,
    unsigned int nsaddrs,
    const struct in_addr saddrs[],
    const struct in_addr smasks[],
    unsigned int ndaddrs,
    const struct in_addr daddrs[],
    const struct in_addr dmask[],
    int verbose,
    struct xtc_handle *handle)
{
    unsigned int i, j;
    int ret = 1;

    for (i = 0; i < nsaddrs; i++) {
        fw->ip.src.s_addr = saddrs[i].s_addr;
        fw->ip.smask.s_addr = smasks[i].s_addr;
        for (j = 0; j < ndaddrs; j++) {
            fw->ip.dst.s_addr = daddrs[j].s_addr;
            fw->ip.dmask.s_addr = dmask[j].s_addr;
            if(verbose)
                print_firewall_line(fw, handle);
            ret &= iptc_insert_entry(chain, fw, rulenum, handle);
        }
    }

    return ret;
}

static unsigned char *
make_delete_mask(const struct xtables_rule_match *matches,
    const struct xtables_target *target)
{
    /* Establish mask for comparison */
    unsigned int size;
    const struct xtables_rule_match *matchp;
    unsigned char *mask, *mptr;

    size = sizeof(struct ipt_entry);
    for (matchp = matches; matchp; matchp = matchp->next)
        size += XT_ALIGN(sizeof(struct xt_entry_match)) + matchp->match->size;

```

```

mask = xtables_calloc(1, size
+ XT_ALIGN(sizeof(struct xt_entry_target))
+ target->size);

memset(mask, 0xFF, sizeof(struct ipt_entry));
mptr = mask + sizeof(struct ipt_entry);

for (matchp = matches; matchp; matchp = matchp->next) {
memset(mptr, 0xFF,
    XT_ALIGN(sizeof(struct xt_entry_match))
    + matchp->match->userspace_size);
mptr += XT_ALIGN(sizeof(struct xt_entry_match)) + matchp->match->size;
}

memset(mptr, 0xFF,
    XT_ALIGN(sizeof(struct xt_entry_target))
    + target->userspace_size);

return mask;
}

static int
delete_entry(const xt_chainlabel chain,
    struct ipt_entry *fw,
    unsigned int nsaddrs,
    const struct in_addr saddrs[],
    const struct in_addr smasks[],
    unsigned int ndaddrs,
    const struct in_addr daddrs[],
    const struct in_addr dmasks[],
    int verbose,
    struct xtc_handle *handle,
    struct xtables_rule_match *matches,
    const struct xtables_target *target)
{
    unsigned int i, j;
    int ret = 1;
    unsigned char *mask;

    mask = make_delete_mask(matches, target);
    for (i = 0; i < nsaddrs; i++) {
        fw->ip.src.s_addr = saddrs[i].s_addr;
        fw->ip.smask.s_addr = smasks[i].s_addr;
        for (j = 0; j < ndaddrs; j++) {
            fw->ip.dst.s_addr = daddrs[j].s_addr;
            fw->ip.dmask.s_addr = dmasks[j].s_addr;
            if (verbose)
                print_firewall_line(fw, handle);
            ret &= iptc_delete_entry(chain, fw, mask, handle);
        }
    }
    free(mask);

    return ret;
}

```

```

}

static int
check_entry(const xt_chainlabel chain, struct ipt_entry *fw,
            unsigned int nsaddrs, const struct in_addr *saddrs,
            const struct in_addr *smasks, unsigned int ndaddrs,
            const struct in_addr *daddrs, const struct in_addr *dmasks,
            bool verbose, struct xtc_handle *handle,
            struct xtables_rule_match *matches,
            const struct xtables_target *target)
{
    unsigned int i, j;
    int ret = 1;
    unsigned char *mask;

    mask = make_delete_mask(matches, target);
    for (i = 0; i < nsaddrs; i++) {
        fw->ip.src.s_addr = saddrs[i].s_addr;
        fw->ip.smask.s_addr = smasks[i].s_addr;
        for (j = 0; j < ndaddrs; j++) {
            fw->ip.dst.s_addr = daddrs[j].s_addr;
            fw->ip.dmask.s_addr = dmasks[j].s_addr;
            if (verbose)
                print_firewall_line(fw, handle);
            ret &= iptc_check_entry(chain, fw, mask, handle);
        }
    }

    free(mask);
    return ret;
}

int
for_each_chain4(int (*fn)(const xt_chainlabel, int, struct xtc_handle *),
               int verbose, int builtinsoo, struct xtc_handle *handle)
{
    int ret = 1;
    const char *chain;
    char *chains;
    unsigned int i, chaincount = 0;

    chain = iptc_first_chain(handle);
    while (chain) {
        chaincount++;
        chain = iptc_next_chain(handle);
    }

    chains = xtables_malloc(sizeof(xt_chainlabel) * chaincount);
    i = 0;
    chain = iptc_first_chain(handle);
    while (chain) {
        strcpy(chains + i*sizeof(xt_chainlabel), chain);
        i++;
        chain = iptc_next_chain(handle);
    }

```

```

    }

    for (i = 0; i < chaincount; i++) {
    if (!builtinstoo
        && iptc_builtin(chains + i*sizeof(xt_chainlabel),
            handle) == 1)
        continue;
        ret &= fn(chains + i*sizeof(xt_chainlabel), verbose, handle);
    }

    free(chains);
    return ret;
}

int
flush_entries4(const xt_chainlabel chain, int verbose,
    struct xtc_handle *handle)
{
    if (!chain)
        return for_each_chain4(flush_entries4, verbose, 1, handle);

    if (verbose)
        fprintf(stdout, "Flushing chain `%s'\n", chain);
    return iptc_flush_entries(chain, handle);
}

static int
zero_entries(const xt_chainlabel chain, int verbose,
    struct xtc_handle *handle)
{
    if (!chain)
        return for_each_chain4(zero_entries, verbose, 1, handle);

    if (verbose)
        fprintf(stdout, "Zeroing chain `%s'\n", chain);
    return iptc_zero_entries(chain, handle);
}

int
delete_chain4(const xt_chainlabel chain, int verbose,
    struct xtc_handle *handle)
{
    if (!chain)
        return for_each_chain4(delete_chain4, verbose, 0, handle);

    if (verbose)
        fprintf(stdout, "Deleting chain `%s'\n", chain);
    return iptc_delete_chain(chain, handle);
}

static int
list_entries(const xt_chainlabel chain, int rulenum, int verbose, int numeric,
    int expanded, int linenumbers, struct xtc_handle *handle)
{
    int found = 0;

```

```
unsigned int format;
const char *this;

format = FMT_OPTIONS;
if (!verbose)
format |= FMT_NOCOUNTS;
else
format |= FMT_VIA;

if (numeric)
format |= FMT_NUMERIC;

if (!expanded)
format |= FMT_KILOMEGAGIGA;

if (linenumbers)
format |= FMT_LINENUMBERS;

for (this = iptc_first_chain(handle);
     this;
     this = iptc_next_chain(handle)) {
const struct ipt_entry *i;
unsigned int num;

if (chain && strcmp(chain, this) != 0)
continue;

if (found) printf("\n");

if (!rulenum)
print_header(format, this, handle);
i = iptc_first_rule(this, handle);

num = 0;
while (i) {
num++;
if (!rulenum || num == rulenum)
print_firewall(i,
               iptc_get_target(i, handle),
               num,
               format,
               handle);
i = iptc_next_rule(i, handle);
}
found = 1;
}

errno = ENOENT;
return found;
}

static void print_proto(uint16_t proto, int invert)
{
if (proto) {
```

```

unsigned int i;
const char *invertstr = invert ? " !" : "";

const struct protoent *pent = getprotobyname(proto);
if (pent) {
    printf("%s -p %s", invertstr, pent->p_name);
    return;
}

for (i = 0; xtables_chain_protos[i].name != NULL; ++i)
    if (xtables_chain_protos[i].num == proto) {
        printf("%s -p %s",
            invertstr, xtables_chain_protos[i].name);
        return;
    }

printf("%s -p %u", invertstr, proto);
}
}

#define IP_PARTS_NATIVE(n)\
(unsigned int)((n)>>24)&0xFF,\
(unsigned int)((n)>>16)&0xFF,\
(unsigned int)((n)>>8)&0xFF,\
(unsigned int)((n)&0xFF)

#define IP_PARTS(n) IP_PARTS_NATIVE(ntohl(n))

/* This assumes that mask is contiguous, and byte-bounded. */
static void
print_iface(char letter, const char *iface, const unsigned char *mask,
    int invert)
{
    unsigned int i;

    if (mask[0] == 0)
        return;

    printf("%s -%c ", invert ? " !" : "", letter);

    for (i = 0; i < IFNAMSIZ; i++) {
        if (mask[i] != 0) {
            if (iface[i] != '\0')
                printf("%c", iface[i]);
        } else {
            /* we can access iface[i-1] here, because
             * a few lines above we make sure that mask[0] != 0 */
            if (iface[i-1] != '\0')
                printf("+");
            break;
        }
    }
}

static int print_match_save(const struct xt_entry_match *e,

```



```

const struct ipt_ip *ip)
{
const struct xtables_match *match =
xtables_find_match(e->u.user.name, XTF_TRY_LOAD, NULL);

if (match) {
printf("-m %s",
match->alias ? match->alias(e) : e->u.user.name);

/* some matches don't provide a save function */
if (match->save)
match->save(ip, e);
} else {
if (e->u.match_size) {
fprintf(stderr,
"Can't find library for match '%s'\n",
e->u.user.name);
exit(1);
}
}
return 0;
}

/* Print a given ip including mask if necessary. */
static void print_ip(const char *prefix, uint32_t ip,
uint32_t mask, int invert)
{
uint32_t bits, hmask = ntohl(mask);
int i;

if (!mask && !ip && !invert)
return;

printf("%s %s %u.%u.%u.%u",
invert ? " !" : "",
prefix,
IP_PARTS(ip));

if (mask == 0xFFFFFFFFU) {
printf("/32");
return;
}

i = 32;
bits = 0xFFFFFFFFU;
while (--i >= 0 && hmask != bits)
bits <<= 1;
if (i >= 0)
printf("/%u", i);
else
printf("/%u.%u.%u.%u", IP_PARTS(mask));
}

/* We want this to be readable, so only print out necessary fields.

```

```

    * Because that's the kind of world I want to live in.
    */
void print_rule4(const struct ipt_entry *e,
struct xtc_handle *h, const char *chain, int counters)
{
    const struct xt_entry_target *t;
    const char *target_name;

    /* print counters for iptables-save */
    if (counters > 0)
        printf("[%llu:%llu] ", (unsigned long long)e->counters.pcnt, (unsigned long long)e->counters.bcmt);

    /* print chain name */
    printf("-A %s", chain);

    /* Print IP part. */
    print_ip("-s", e->ip.src.s_addr, e->ip.smask.s_addr,
e->ip.invflags & IPT_INV_SRCIP);

    print_ip("-d", e->ip.dst.s_addr, e->ip.dmask.s_addr,
e->ip.invflags & IPT_INV_DSTIP);

    print_iface('i', e->ip.iniface, e->ip.iniface_mask,
e->ip.invflags & IPT_INV_VIA_IN);

    print_iface('o', e->ip.outiface, e->ip.outiface_mask,
e->ip.invflags & IPT_INV_VIA_OUT);

    print_proto(e->ip.proto, e->ip.invflags & XT_INV_PROTO);

    if (e->ip.flags & IPT_F_FRAG)
        printf("%s -f",
e->ip.invflags & IPT_INV_FRAG ? " !" : "");

    /* Print matchinfo part */
    if (e->target_offset) {
        IPT_MATCH_ITERATE(e, print_match_save, &e->ip);
    }

    /* print counters for iptables -R */
    if (counters < 0)
        printf(" -c %llu %llu", (unsigned long long)e->counters.pcnt, (unsigned long long)e->counters.bcmt);

    /* Print target name and targinfo part */
    target_name = iptc_get_target(e, h);
    t = ipt_get_target((struct ipt_entry *)e);
    if (t->u.user.name[0]) {
        const struct xtables_target *target =
xtables_find_target(t->u.user.name, XTF_TRY_LOAD);

        if (!target) {
            fprintf(stderr, "Can't find library for target '%s'\n",
t->u.user.name);
            exit(1);
        }
    }

```

```

printf(" -j %s", target->alias ? target->alias(t) : target_name);
if (target->save)
target->save(&e->ip, t);
else {
/* If the target size is greater than xt_entry_target
 * there is something to be saved, we just don't know
 * how to print it */
if (t->u.target_size !=
    sizeof(struct xt_entry_target)) {
fprintf(stderr, "Target '%s' is missing "
"save function\n",
t->u.user.name);
exit(1);
}
}
} else if (target_name && (*target_name != '\0'))
#ifdef IPT_F_GOTO
printf(" -%c %s", e->ip.flags & IPT_F_GOTO ? 'g' : 'j', target_name);
#else
printf(" -j %s", target_name);
#endif

printf("\n");
}

static int
list_rules(const xt_chainlabel chain, int rulenum, int counters,
    struct xtc_handle *handle)
{
const char *this = NULL;
int found = 0;

if (counters)
counters = -1; /* iptables -c format */

/* Dump out chain names first,
 * thereby preventing dependency conflicts */
if (!rulenum) for (this = iptc_first_chain(handle);
    this;
    this = iptc_next_chain(handle)) {
if (chain && strcmp(this, chain) != 0)
continue;

if (iptc_builtin(this, handle)) {
struct xt_counters count;
printf("-P %s %s", this, iptc_get_policy(this, &count, handle));
if (counters)
printf(" -c %llu %llu", (unsigned long long)count.pcnt, (unsigned long long)count.bcnt);
printf("\n");
} else {
printf("-N %s\n", this);
}
}
}

```

```

for (this = iptc_first_chain(handle);
    this;
    this = iptc_next_chain(handle)) {
const struct ipt_entry *e;
int num = 0;

if (chain && strcmp(this, chain) != 0)
continue;

/* Dump out rules */
e = iptc_first_rule(this, handle);
while(e) {
num++;
if (!rulenum || num == rulenum)
    print_rule4(e, handle, this, counters);
e = iptc_next_rule(e, handle);
}
found = 1;
}

errno = ENOENT;
return found;
}

static struct ipt_entry *
generate_entry(const struct ipt_entry *fw,
              struct xtables_rule_match *matches,
              struct xt_entry_target *target)
{
unsigned int size;
struct xtables_rule_match *matchp;
struct ipt_entry *e;

size = sizeof(struct ipt_entry);
for (matchp = matches; matchp; matchp = matchp->next)
size += matchp->match->m->u.match_size;

e = xtables_malloc(size + target->u.target_size);
*e = *fw;
e->target_offset = size;
e->next_offset = size + target->u.target_size;

size = 0;
for (matchp = matches; matchp; matchp = matchp->next) {
memcpy(e->elems + size, matchp->match->m, matchp->match->m->u.match_size);
size += matchp->match->m->u.match_size;
}
memcpy(e->elems + size, target, target->u.target_size);

return e;
}

static void command_jump(struct iptables_command_state *cs)
{

```

```

size_t size;

set_option(&cs->options, OPT_JUMP, &cs->fw.ip.invflags, cs->invert);
cs->jumpto = parse_target(optarg);
/* TRY_LOAD (may be chain name) */
cs->target = xtables_find_target(cs->jumpto, XTF_TRY_LOAD);

if (cs->target == NULL)
return;

size = XT_ALIGN(sizeof(struct xt_entry_target))
+ cs->target->size;

cs->target->t = xtables_calloc(1, size);
cs->target->t->u.target_size = size;
if (cs->target->real_name == NULL) {
strcpy(cs->target->t->u.user.name, cs->jumpto);
} else {
/* Alias support for userspace side */
strcpy(cs->target->t->u.user.name, cs->target->real_name);
if (!(cs->target->ext_flags & XTABLES_EXT_ALIAS))
fprintf(stderr, "Notice: The %s target is converted into %s target "
               "in rule listing and saving.\n",
               cs->jumpto, cs->target->real_name);
}
cs->target->t->u.user.revision = cs->target->revision;

xs_init_target(cs->target);

if (cs->target->x6_options != NULL)
opts = xtables_options_xfrm(iptables_globals.orig_opts, opts,
                           cs->target->x6_options,
                           &cs->target->option_offset);
else
opts = xtables_merge_options(iptables_globals.orig_opts, opts,
                           cs->target->extra_opts,
                           &cs->target->option_offset);
if (opts == NULL)
xtables_error(OTHER_PROBLEM, "can't alloc memory!");
}

static void command_match(struct iptables_command_state *cs)
{
struct xtables_match *m;
size_t size;

if (cs->invert)
xtables_error(PARAMETER_PROBLEM,
               "unexpected ! flag before --match");

m = xtables_find_match(optarg, XTF_LOAD_MUST_SUCCEED, &cs->matches);
size = XT_ALIGN(sizeof(struct xt_entry_match)) + m->size;
m->m = xtables_calloc(1, size);
m->m->u.match_size = size;

```

```

if (m->real_name == NULL) {
strcpy(m->m->u.user.name, m->name);
} else {
strcpy(m->m->u.user.name, m->real_name);
if (!(m->ext_flags & XTABLES_EXT_ALIAS))
fprintf(stderr, "Notice: the %s match is converted into %s match "
    "in rule listing and saving.\n", m->name, m->real_name);
}
m->m->u.user.revision = m->revision;

xs_init_match(m);
if (m == m->next)
return;
/* Merge options for non-cloned matches */
if (m->x6_options != NULL)
opts = xtables_options_xfrm(iptables_globals.orig_opts, opts,
    m->x6_options, &m->option_offset);
else if (m->extra_opts != NULL)
opts = xtables_merge_options(iptables_globals.orig_opts, opts,
    m->extra_opts, &m->option_offset);
if (opts == NULL)
xtables_error(OTHER_PROBLEM, "can't alloc memory!");
}

int do_command4(int argc, char *argv[], char **table,
struct xtc_handle **handle, bool restore)
{
struct iptables_command_state cs;
struct ipt_entry *e = NULL;
unsigned int nsaddrs = 0, ndaddrs = 0;
struct in_addr *saddrs = NULL, *smasks = NULL;
struct in_addr *daddrs = NULL, *dmasks = NULL;

int verbose = 0;
int wait = 0;
const char *chain = NULL;
const char *shostnetworkmask = NULL, *dhostnetworkmask = NULL;
const char *policy = NULL, *newname = NULL;
unsigned int rulenum = 0, command = 0;
const char *pcent = NULL, *bcnt = NULL;
int ret = 1;
struct xtables_match *m;
struct xtables_rule_match *matchp;
struct xtables_target *t;
unsigned long long cnt;

memset(&cs, 0, sizeof(cs));
cs.jumpto = "";
cs.argv = argv;

/* re-set optind to 0 in case do_command4 gets called
 * a second time */
optind = 0;

/* clear mflags in case do_command4 gets called a second time

```

```

    * (we clear the global list of all matches for security)*/
    for (m = xtables_matches; m; m = m->next)
        m->mflags = 0;

    for (t = xtables_targets; t; t = t->next) {
        t->tflags = 0;
        t->used = 0;
    }

    /* Suppress error messages: we may add new options if we
       demand-load a protocol. */
    opterr = 0;
    opts = xt_params->orig_opts;
    while ((cs.c = getopt_long(argc, argv,
        "-:A:C:D:R:I:L::S::M:F::Z::N:X::E:P:Vh::o:p:s:d:j:i:fbvw::nt:m:xc:g:46",
        opts, NULL)) != -1) {
        switch (cs.c) {
            /*
             * Command selection
             */
            case 'A':
                add_command(&command, CMD_APPEND, CMD_NONE,
                    cs.invert);
                chain = optarg;
                break;

            case 'C':
                add_command(&command, CMD_CHECK, CMD_NONE,
                    cs.invert);
                chain = optarg;
                break;

            case 'D':
                add_command(&command, CMD_DELETE, CMD_NONE,
                    cs.invert);
                chain = optarg;
                if (optind < argc && argv[optind][0] != '-'
                    && argv[optind][0] != '!') {
                    rulenum = parse_rulenum(argv[optind++]);
                    command = CMD_DELETE_NUM;
                }
                break;

            case 'R':
                add_command(&command, CMD_REPLACE, CMD_NONE,
                    cs.invert);
                chain = optarg;
                if (optind < argc && argv[optind][0] != '-'
                    && argv[optind][0] != '!')
                    rulenum = parse_rulenum(argv[optind++]);
                else
                    xtables_error(PARAMETER_PROBLEM,
                        "-%c requires a rule number",
                        cmd2char(CMD_REPLACE));

```

```

break;

case 'I':
add_command(&command, CMD_INSERT, CMD_NONE,
    cs.invert);
chain = optarg;
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    rulenum = parse_rulenum(argv[optind++]);
else rulenum = 1;
break;

case 'L':
add_command(&command, CMD_LIST,
    CMD_ZERO | CMD_ZERO_NUM, cs.invert);
if (optarg) chain = optarg;
else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    chain = argv[optind++];
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    rulenum = parse_rulenum(argv[optind++]);
break;

case 'S':
add_command(&command, CMD_LIST_RULES,
    CMD_ZERO|CMD_ZERO_NUM, cs.invert);
if (optarg) chain = optarg;
else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    chain = argv[optind++];
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    rulenum = parse_rulenum(argv[optind++]);
break;

case 'F':
add_command(&command, CMD_FLUSH, CMD_NONE,
    cs.invert);
if (optarg) chain = optarg;
else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    chain = argv[optind++];
break;

case 'Z':
add_command(&command, CMD_ZERO, CMD_LIST|CMD_LIST_RULES,
    cs.invert);
if (optarg) chain = optarg;
else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
    chain = argv[optind++];
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!') {
    rulenum = parse_rulenum(argv[optind++]);
}

```



```
command = CMD_ZERO_NUM;
}
break;

case 'N':
parse_chain(optarg);
add_command(&command, CMD_NEW_CHAIN, CMD_NONE,
    cs.invert);
chain = optarg;
break;

case 'X':
add_command(&command, CMD_DELETE_CHAIN, CMD_NONE,
    cs.invert);
if (optarg) chain = optarg;
else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
chain = argv[optind++];
break;

case 'E':
add_command(&command, CMD_RENAME_CHAIN, CMD_NONE,
    cs.invert);
chain = optarg;
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
newname = argv[optind++];
else
xtables_error(PARAMETER_PROBLEM,
    "-%c requires old-chain-name and "
    "new-chain-name",
    cmd2char(CMD_RENAME_CHAIN));
break;

case 'P':
add_command(&command, CMD_SET_POLICY, CMD_NONE,
    cs.invert);
chain = optarg;
if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
policy = argv[optind++];
else
xtables_error(PARAMETER_PROBLEM,
    "-%c requires a chain and a policy",
    cmd2char(CMD_SET_POLICY));
break;

case 'h':
if (!optarg)
optarg = argv[optind];

/* iptables -p icmp -h */
if (!cs.matches && cs.protocol)
xtables_find_match(cs.protocol,
```

```

XTF_TRY_LOAD, &cs.matches);

exit_printhelp(cs.matches);

/*
 * Option selection
 */
case 'p':
set_option(&cs.options, OPT_PROTOCOL, &cs.fw.ip.invflags,
    cs.invert);

/* Canonicalize into lower case */
for (cs.protocol = optarg; *cs.protocol; cs.protocol++)
*cs.protocol = tolower(*cs.protocol);

cs.protocol = optarg;
cs.fw.ip.proto = xtables_parse_protocol(cs.protocol);

if (cs.fw.ip.proto == 0
    && (cs.fw.ip.invflags & XT_INV_PROTO))
xtables_error(PARAMETER_PROBLEM,
    "rule would never match protocol");
break;

case 's':
set_option(&cs.options, OPT_SOURCE, &cs.fw.ip.invflags,
    cs.invert);
shostnetworkmask = optarg;
break;

case 'd':
set_option(&cs.options, OPT_DESTINATION, &cs.fw.ip.invflags,
    cs.invert);
dhostnetworkmask = optarg;
break;

#ifdef IPT_F_GOTO
case 'g':
set_option(&cs.options, OPT_JUMP, &cs.fw.ip.invflags,
    cs.invert);
cs.fw.ip.flags |= IPT_F_GOTO;
cs.jumpton = parse_target(optarg);
break;
#endif

case 'j':
command_jump(&cs);
break;

case 'i':
if (*optarg == '\0')
xtables_error(PARAMETER_PROBLEM,
    "Empty interface is likely to be "
    "undesired");

```

```
set_option(&cs.options, OPT_VIANAMEIN, &cs.fw.ip.invflags,
    cs.invert);
xtables_parse_interface(optarg,
cs.fw.ip.iniface,
cs.fw.ip.iniface_mask);
break;

case 'o':
if (*optarg == '\0')
xtables_error(PARAMETER_PROBLEM,
"Empty interface is likely to be "
"undesired");
set_option(&cs.options, OPT_VIANAMEOUT, &cs.fw.ip.invflags,
    cs.invert);
xtables_parse_interface(optarg,
cs.fw.ip.outiface,
cs.fw.ip.outiface_mask);
break;

case 'f':
set_option(&cs.options, OPT_FRAGMENT, &cs.fw.ip.invflags,
    cs.invert);
cs.fw.ip.flags |= IPT_F_FRAG;
break;

case 'v':
if (!verbose)
set_option(&cs.options, OPT_VERBOSE,
    &cs.fw.ip.invflags, cs.invert);
verbose++;
break;

case 'w':
if (restore) {
xtables_error(PARAMETER_PROBLEM,
    "You cannot use '-w' from "
    "iptables-restore");
}
wait = -1;
if (optarg) {
if (sscanf(optarg, "%i", &wait) != 1)
xtables_error(PARAMETER_PROBLEM,
"wait seconds not numeric");
} else if (optind < argc && argv[optind][0] != '-'
    && argv[optind][0] != '!')
if (sscanf(argv[optind++], "%i", &wait) != 1)
xtables_error(PARAMETER_PROBLEM,
"wait seconds not numeric");
break;

case 'm':
command_match(&cs);
break;
```

```

case 'n':
set_option(&cs.options, OPT_NUMERIC, &cs.fw.ip.invflags,
cs.invert);
break;

case 't':
if (cs.invert)
xtables_error(PARAMETER_PROBLEM,
"unexpected ! flag before --table");
*table = optarg;
break;

case 'x':
set_option(&cs.options, OPT_EXPANDED, &cs.fw.ip.invflags,
cs.invert);
break;

case 'V':
if (cs.invert)
printf("Not %s ;-\n", prog_vers);
else
printf("%s v%s\n",
prog_name, prog_vers);
exit(0);

case 'O':
set_option(&cs.options, OPT_LINENUMBERS, &cs.fw.ip.invflags,
cs.invert);
break;

case 'M':
xtables_modprobe_program = optarg;
break;

case 'c':

set_option(&cs.options, OPT_COUNTERS, &cs.fw.ip.invflags,
cs.invert);
pcnt = optarg;
bcnt = strchr(pcnt + 1, ',');
if (bcnt)
bcnt++;
if (!bcnt && optind < argc && argv[optind][0] != '-'
&& argv[optind][0] != '!')
bcnt = argv[optind++];
if (!bcnt)
xtables_error(PARAMETER_PROBLEM,
"-%c requires packet and byte counter",
opt2char(OPT_COUNTERS));

if (sscanf(pcnt, "%llu", &cnt) != 1)
xtables_error(PARAMETER_PROBLEM,
"-%c packet counter not numeric",
opt2char(OPT_COUNTERS));
cs.fw.counters.pcnt = cnt;

```

```

if (sscanf(bcnt, "%llu", &cnt) != 1)
xtables_error(PARAMETER_PROBLEM,
"-%c byte counter not numeric",
opt2char(OPT_COUNTERS));
cs.fw.counters.bcnt = cnt;
break;

case '4':
/* This is indeed the IPv4 iptables */
break;

case '6':
/* This is not the IPv6 ip6tables */
if (line != -1)
return 1; /* success: line ignored */
fprintf(stderr, "This is the IPv4 version of iptables.\n");
exit_tryhelp(2);

case 1: /* non option */
if (optarg[0] == '!' && optarg[1] == '\0') {
if (cs.invert)
xtables_error(PARAMETER_PROBLEM,
"multiple consecutive ! not"
" allowed");
cs.invert = TRUE;
optarg[0] = '\0';
continue;
}
fprintf(stderr, "Bad argument '%s'\n", optarg);
exit_tryhelp(2);

default:
if (command_default(&cs, &iptables_globals) == 1)
/* cf. ip6tables.c */
continue;
break;
}
cs.invert = FALSE;
}

if (strcmp(*table, "nat") == 0 &&
((policy != NULL && strcmp(policy, "DROP") == 0) ||
(cs.jumpton != NULL && strcmp(cs.jumpton, "DROP") == 0)))
xtables_error(PARAMETER_PROBLEM,
"\n\nThe \"%s\" table is not intended for filtering, "
"the use of DROP is therefore inhibited.\n\n");

for (matchp = cs.matches; matchp; matchp = matchp->next)
xtables_option_mfcall(matchp->match);
if (cs.target != NULL)
xtables_option_tfcall(cs.target);

/* Fix me: must put inverse options checking here --MN */

```

```

if (optind < argc)
xtables_error(PARAMETER_PROBLEM,
    "unknown arguments found on commandline");
if (!command)
xtables_error(PARAMETER_PROBLEM, "no command specified");
if (cs.invert)
xtables_error(PARAMETER_PROBLEM,
    "nothing appropriate following !");

if (command & (CMD_REPLACE | CMD_INSERT | CMD_DELETE | CMD_APPEND | CMD_CHECK)) {
if (!(cs.options & OPT_DESTINATION))
dhostnetworkmask = "0.0.0.0/0";
if (!(cs.options & OPT_SOURCE))
shostnetworkmask = "0.0.0.0/0";
}

if (shostnetworkmask)
xtables_ipparse_multiple(shostnetworkmask, &saddrs,
    &smasks, &nsaddrs);

if (dhostnetworkmask)
xtables_ipparse_multiple(dhostnetworkmask, &daddrs,
    &dmasks, &ndaddrs);

if ((nsaddrs > 1 || ndaddrs > 1) &&
    (cs.fw.ip.inflags & (IPT_INV_SRCIP | IPT_INV_DSTIP)))
xtables_error(PARAMETER_PROBLEM, "! not allowed with multiple"
    " source or destination IP addresses");

if (command == CMD_REPLACE && (nsaddrs != 1 || ndaddrs != 1))
xtables_error(PARAMETER_PROBLEM, "Replacement rule does not "
    "specify a unique address");

generic_opt_check(command, cs.options);

/* Attempt to acquire the xtables lock */
if (!restore && !xtables_lock(wait)) {
fprintf(stderr, "Another app is currently holding the xtables lock. ");
if (wait == 0)
fprintf(stderr, "Perhaps you want to use the -w option?\n");
else
fprintf(stderr, "Stopped waiting after %ds.\n", wait);
xtables_free_opts(1);
exit(RESOURCE_PROBLEM);
}

/* only allocate handle if we weren't called with a handle */
if (!*handle)
*handle = iptc_init(*table);

/* try to insmod the module if iptc_init failed */
if (!*handle && xtables_load_ko(xtables_modprobe_program, false) != -1)
*handle = iptc_init(*table);

```

```

if (!*handle)
xtables_error(VERSION_PROBLEM,
    "can't initialize iptables table '%s': %s",
    *table, iptc_strerror(errno));

if (command == CMD_APPEND
    || command == CMD_DELETE
    || command == CMD_CHECK
    || command == CMD_INSERT
    || command == CMD_REPLACE) {
if (strcmp(chain, "PREROUTING") == 0
    || strcmp(chain, "INPUT") == 0) {
/* -o not valid with incoming packets. */
if (cs.options & OPT_VIANAMEOUT)
xtables_error(PARAMETER_PROBLEM,
    "Can't use -%c with %s\n",
    opt2char(OPT_VIANAMEOUT),
    chain);
}

if (strcmp(chain, "POSTROUTING") == 0
    || strcmp(chain, "OUTPUT") == 0) {
/* -i not valid with outgoing packets */
if (cs.options & OPT_VIANAMEIN)
xtables_error(PARAMETER_PROBLEM,
    "Can't use -%c with %s\n",
    opt2char(OPT_VIANAMEIN),
    chain);
}

if (cs.target && iptc_is_chain(cs.jumpto, *handle)) {
fprintf(stderr,
    "Warning: using chain %s, not extension\n",
    cs.jumpto);

if (cs.target->t)
free(cs.target->t);

cs.target = NULL;
}

/* If they didn't specify a target, or it's a chain
   name, use standard. */
if (!cs.target
    && (strlen(cs.jumpto) == 0
    || iptc_is_chain(cs.jumpto, *handle))) {
size_t size;

cs.target = xtables_find_target(XT_STANDARD_TARGET,
    XTF_LOAD_MUST_SUCCEED);

size = sizeof(struct xt_entry_target)
    + cs.target->size;
cs.target->t = xtables_calloc(1, size);

```

```

cs.target->t->u.target_size = size;
strcpy(cs.target->t->u.user.name, cs.jumpton);
if (!iptc_is_chain(cs.jumpton, *handle))
cs.target->t->u.user.revision = cs.target->revision;
xs_init_target(cs.target);
}

if (!cs.target) {
/* It is no chain, and we can't load a plugin.
 * We cannot know if the plugin is corrupt, non
 * existent OR if the user just misspelled a
 * chain.
 */
#ifdef IPT_F_GOTO
if (cs.fw.ip.flags & IPT_F_GOTO)
xtables_error(PARAMETER_PROBLEM,
    "goto '%s' is not a chain\n",
    cs.jumpton);
#endif
xtables_find_target(cs.jumpton, XTF_LOAD_MUST_SUCCEED);
} else {
e = generate_entry(&cs.fw, cs.matches, cs.target->t);
free(cs.target->t);
}
}

switch (command) {
case CMD_APPEND:
ret = append_entry(chain, e,
    nsaddrs, saddrs, smasks,
    ndaddrs, daddrs, dmask,
    cs.options&OPT_VERBOSE,
    *handle);
break;
case CMD_DELETE:
ret = delete_entry(chain, e,
    nsaddrs, saddrs, smasks,
    ndaddrs, daddrs, dmask,
    cs.options&OPT_VERBOSE,
    *handle, cs.matches, cs.target);
break;
case CMD_DELETE_NUM:
ret = iptc_delete_num_entry(chain, rulenum - 1, *handle);
break;
case CMD_CHECK:
ret = check_entry(chain, e,
    nsaddrs, saddrs, smasks,
    ndaddrs, daddrs, dmask,
    cs.options&OPT_VERBOSE,
    *handle, cs.matches, cs.target);
break;
case CMD_REPLACE:
ret = replace_entry(chain, e, rulenum - 1,
    saddrs, smasks, daddrs, dmask,
    cs.options&OPT_VERBOSE, *handle);

```



```
break;
case CMD_INSERT:
ret = insert_entry(chain, e, rulenum - 1,
    nsaddrs, saddrs, smasks,
    ndaddrs, daddrs, dmask,
    cs.options&OPT_VERBOSE,
    *handle);
break;
case CMD_FLUSH:
ret = flush_entries4(chain, cs.options&OPT_VERBOSE, *handle);
break;
case CMD_ZERO:
ret = zero_entries(chain, cs.options&OPT_VERBOSE, *handle);
break;
case CMD_ZERO_NUM:
ret = iptc_zero_counter(chain, rulenum, *handle);
break;
case CMD_LIST:
case CMD_LIST|CMD_ZERO:
case CMD_LIST|CMD_ZERO_NUM:
ret = list_entries(chain,
    rulenum,
    cs.options&OPT_VERBOSE,
    cs.options&OPT_NUMERIC,
    cs.options&OPT_EXPANDED,
    cs.options&OPT_LINENUMBERS,
    *handle);
if (ret && (command & CMD_ZERO))
ret = zero_entries(chain,
    cs.options&OPT_VERBOSE, *handle);
if (ret && (command & CMD_ZERO_NUM))
ret = iptc_zero_counter(chain, rulenum, *handle);
break;
case CMD_LIST_RULES:
case CMD_LIST_RULES|CMD_ZERO:
case CMD_LIST_RULES|CMD_ZERO_NUM:
ret = list_rules(chain,
    rulenum,
    cs.options&OPT_VERBOSE,
    *handle);
if (ret && (command & CMD_ZERO))
ret = zero_entries(chain,
    cs.options&OPT_VERBOSE, *handle);
if (ret && (command & CMD_ZERO_NUM))
ret = iptc_zero_counter(chain, rulenum, *handle);
break;
case CMD_NEW_CHAIN:
ret = iptc_create_chain(chain, *handle);
break;
case CMD_DELETE_CHAIN:
ret = delete_chain4(chain, cs.options&OPT_VERBOSE, *handle);
break;
case CMD_RENAME_CHAIN:
ret = iptc_rename_chain(chain, newname, *handle);
```

```
break;
case CMD_SET_POLICY:
ret = iptc_set_policy(chain, policy, cs.options&OPT_COUNTERS ? &cs.fw.counters : NULL, *handle);
break;
default:
/* We should never reach this... */
exit_tryhelp(2);
}

if (verbose > 1)
dump_entries(*handle);

xtables_rule_matches_free(&cs.matches);

if (e != NULL) {
free(e);
e = NULL;
}

free(saddrs);
free(smasks);
free(daddrs);
free(dmasks);
xtables_free_opts(1);

return ret;
}
```

1.55 iputils

1.55.1 Available under license

```
/*
 * arping.c
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version
 * 2 of the License, or (at your option) any later version.
 *
 * Authors: Alexey Kuznetsov, <kuznet@ms2.inr.ac.ru>
 * YOSHIFUJI Hideaki <yoshfuji@linux-ipv6.org>
 */

#include <stdlib.h>
#include <sys/param.h>
#include <sys/socket.h>
#include <linux/sockios.h>
#include <sys/file.h>
#include <sys/time.h>
#include <sys/signal.h>
#include <sys/ioctl.h>
#include <net/if.h>
#include <linux/if_packet.h>
#include <linux/if_ether.h>
#include <net/if_arp.h>
#include <sys/uio.h>
#ifdef CAPABILITIES
#include <sys/prctl.h>
#include <sys/capability.h>
#endif

#include <netdb.h>
#include <unistd.h>
#include <stdio.h>
#include <ctype.h>
#include <errno.h>
#include <string.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#ifdef USE_SYSFS
#include <sysfs/libsysfs.h>
struct sysfs_devattr_values;
#endif

#ifdef WITHOUT_IFADDRS
#include <ifaddrs.h>
#endif
```

```

#ifndef USE_IDN
#include <idna.h>
#include <locale.h>
#endif

#include "SNAPSHOT.h"

static void usage(void) __attribute__((noreturn));

#ifndef DEFAULT_DEVICE
# define DEFAULT_DEVICE_STRDEFAULT_DEVICE
#else
# define DEFAULT_DEVICE NULL
#endif

struct device {
char *name;
int ifindex;
#ifndef WITHOUT_IFADDRS
struct ifaddrs *ifa;
#endif
#ifdef USE_SYSFS
struct sysfs_devattr_values *sysfs;
#endif
};

int quit_on_reply=0;
struct device device = {
.name = DEFAULT_DEVICE,
};
char *source;
struct in_addr src, dst;
char *target;
int dad, unsolicited, advert;
int quiet;
int count=-1;
int timeout;
int unicasting;
int s;
int broadcast_only;

struct sockaddr_storage me;
struct sockaddr_storage he;

struct timeval start, last;

int sent, brd_sent;
int received, brd_rcv, req_rcv;

#ifndef CAPABILITIES
static uid_t euid;
#endif

#define MS_TDIFF(tv1,tv2) ( ((tv1).tv_sec-(tv2).tv_sec)*1000 + \

```

```

((tv1).tv_usec-(tv2).tv_usec)/1000 )

#define OFFSET_OF(name,ele)((size_t)(((name *)0)->ele))

static inline socklen_t sll_len(size_t halen)
{
    socklen_t len = OFFSET_OF(struct sockaddr_ll, sll_addr) + halen;
    if (len < sizeof(struct sockaddr_ll))
        len = sizeof(struct sockaddr_ll);
    return len;
}

#define SLL_LEN(hln)sll_len(hln)

void usage(void)
{
    fprintf(stderr,
        "Usage: arping [-fqbDUAV] [-c count] [-w timeout] [-I device] [-s source] destination\n"
        " -f : quit on first reply\n"
        " -q : be quiet\n"
        " -b : keep broadcasting, don't go unicast\n"
        " -D : duplicate address detection mode\n"
        " -U : Unsolicited ARP mode, update your neighbours\n"
        " -A : ARP answer mode, update your neighbours\n"
        " -V : print version and exit\n"
        " -c count : how many packets to send\n"
        " -w timeout : how long to wait for a reply\n"
        " -I device : which ethernet device to use"
#ifdef DEFAULT_DEVICE_STR
        " (" DEFAULT_DEVICE_STR ") "
#endif
        "\n"
        " -s source : source ip address\n"
        " destination : ask for what ip address\n"
    );
    exit(2);
}

void set_signal(int signo, void (*handler)(void))
{
    struct sigaction sa;

    memset(&sa, 0, sizeof(sa));
    sa.sa_handler = (void (*)(int))handler;
    sa.sa_flags = SA_RESTART;
    sigaction(signo, &sa, NULL);
}

#ifdef CAPABILITIES
static const cap_value_t caps[] = { CAP_NET_RAW, };
static cap_flag_value_t cap_raw = CAP_CLEAR;
#endif

void limit_capabilities(void)

```

```

{
#ifdef CAPABILITIES
cap_t cap_p;

cap_p = cap_get_proc();
if (!cap_p) {
perror("arping: cap_get_proc");
exit(-1);
}

cap_get_flag(cap_p, CAP_NET_RAW, CAP_PERMITTED, &cap_raw);

if (cap_raw != CAP_CLEAR) {
if (cap_clear(cap_p) < 0) {
perror("arping: cap_clear");
exit(-1);
}

cap_set_flag(cap_p, CAP_PERMITTED, 1, caps, CAP_SET);

if (cap_set_proc(cap_p) < 0) {
perror("arping: cap_set_proc");
if (errno != EPERM)
exit(-1);
}
}

if (prctl(PR_SET_KEEPCAPS, 1) < 0) {
perror("arping: prctl");
exit(-1);
}

if (setuid(getuid()) < 0) {
perror("arping: setuid");
exit(-1);
}

if (prctl(PR_SET_KEEPCAPS, 0) < 0) {
perror("arping: prctl");
exit(-1);
}

cap_free(cap_p);
#else
euid = geteuid();
#endif
}

int modify_capability_raw(int on)
{
#ifdef CAPABILITIES
cap_t cap_p;

if (cap_raw != CAP_SET)
return on ? -1 : 0;

```

```
cap_p = cap_get_proc();
if (!cap_p) {
    perror("arping: cap_get_proc");
    return -1;
}

cap_set_flag(cap_p, CAP_EFFECTIVE, 1, caps, on ? CAP_SET : CAP_CLEAR);

if (cap_set_proc(cap_p) < 0) {
    perror("arping: cap_set_proc");
    return -1;
}

cap_free(cap_p);
#else
if (setuid(on ? euid : getuid())) {
    perror("arping: setuid");
    return -1;
}
#endif
return 0;
}

static inline int enable_capability_raw(void)
{
    return modify_capability_raw(1);
}

static inline int disable_capability_raw(void)
{
    return modify_capability_raw(0);
}

void drop_capabilities(void)
{
    #ifdef CAPABILITIES
    cap_t cap_p = cap_init();

    if (!cap_p) {
        perror("arping: cap_init");
        exit(-1);
    }

    if (cap_set_proc(cap_p) < 0) {
        perror("arping: cap_set_proc");
        exit(-1);
    }

    cap_free(cap_p);
    #else
    if (setuid(getuid()) < 0) {
        perror("arping: setuid");
        exit(-1);
    }
    #endif
}
```

```

    }
#endif
}

int send_pack(int s, struct in_addr src, struct in_addr dst,
              struct sockaddr_ll *ME, struct sockaddr_ll *HE)
{
    int err;
    struct timeval now;
    unsigned char buf[256];
    struct arphdr *ah = (struct arphdr*)buf;
    unsigned char *p = (unsigned char*)(ah+1);

    ah->ar_hrd = htons(ME->sll_hatype);
    if (ah->ar_hrd == htons(ARPHRD_FDDI))
        ah->ar_hrd = htons(ARPHRD_ETHER);
    ah->ar_pro = htons(ETH_P_IP);
    ah->ar_hln = ME->sll_halen;
    ah->ar_pln = 4;
    ah->ar_op = advert ? htons(ARPOP_REPLY) : htons(ARPOP_REQUEST);

    memcpy(p, &ME->sll_addr, ah->ar_hln);
    p+=ME->sll_halen;

    memcpy(p, &src, 4);
    p+=4;

    if (advert)
        memcpy(p, &ME->sll_addr, ah->ar_hln);
    else
        memcpy(p, &HE->sll_addr, ah->ar_hln);
    p+=ah->ar_hln;

    memcpy(p, &dst, 4);
    p+=4;

    gettimeofday(&now, NULL);
    err = sendto(s, buf, p-buf, 0, (struct sockaddr*)HE, SLL_LEN(ah->ar_hln));
    if (err == p-buf) {
        last = now;
        sent++;
        if (!unicasting)
            brd_sent++;
    }
    return err;
}

void finish(void)
{
    if (!quiet) {
        printf("Sent %d probes (%d broadcast(s))\n", sent, brd_sent);
        printf("Received %d response(s)", received);
        if (brd_rcv || req_rcv) {
            printf(" ");
            if (req_rcv)

```



```

printf("%d request(s)", req_rcv);
if (brd_rcv)
printf("%s%d broadcast(s)",
    req_rcv ? ", " : "",
    brd_rcv);
printf("");
}
printf("\n");
fflush(stdout);
}
if (dad)
exit(!received);
if (unsolicited)
exit(0);
exit(!received);
}

void catcher(void)
{
struct timeval tv, tv_s, tv_o;

gettimeofday(&tv, NULL);

if (start.tv_sec==0)
start = tv;

timersub(&tv, &start, &tv_s);
tv_o.tv_sec = timeout;
tv_o.tv_usec = 500 * 1000;

if (count-- == 0 || (timeout && timercmp(&tv_s, &tv_o, >)))
finish();

timersub(&tv, &last, &tv_s);
tv_o.tv_sec = 0;

if (last.tv_sec==0 || timercmp(&tv_s, &tv_o, >)) {
send_pack(s, src, dst,
    (struct sockaddr_ll *)&me, (struct sockaddr_ll *)&he);
if (count == 0 && unsolicited)
finish();
}
alarm(1);
}

void print_hex(unsigned char *p, int len)
{
int i;
for (i=0; i<len; i++) {
printf("%02X", p[i]);
if (i != len-1)
printf(":");
}
}

```

```

int recv_pack(unsigned char *buf, int len, struct sockaddr_ll *FROM)
{
    struct timeval tv;
    struct arphdr *ah = (struct arphdr*)buf;
    unsigned char *p = (unsigned char *)(ah+1);
    struct in_addr src_ip, dst_ip;

    gettimeofday(&tv, NULL);

    /* Filter out wild packets */
    if (FROM->sll_pkttype != PACKET_HOST &&
        FROM->sll_pkttype != PACKET_BROADCAST &&
        FROM->sll_pkttype != PACKET_MULTICAST)
        return 0;

    /* Only these types are recognised */
    if (ah->ar_op != htons(ARPOP_REQUEST) &&
        ah->ar_op != htons(ARPOP_REPLY))
        return 0;

    /* ARPHRD check and this darned FDDI hack here :-( */
    if (ah->ar_hrd != htons(FROM->sll_hatype) &&
        (FROM->sll_hatype != ARPHRD_FDDI || ah->ar_hrd != htons(ARPHRD_ETHER)))
        return 0;

    /* Protocol must be IP. */
    if (ah->ar_pro != htons(ETH_P_IP))
        return 0;
    if (ah->ar_pln != 4)
        return 0;
    if (ah->ar_hln != ((struct sockaddr_ll *)&me)->sll_halen)
        return 0;
    if (len < sizeof(*ah) + 2*(4 + ah->ar_hln))
        return 0;
    memcpy(&src_ip, p+ah->ar_hln, 4);
    memcpy(&dst_ip, p+ah->ar_hln+4+ah->ar_hln, 4);
    if (!dad) {
        if (src_ip.s_addr != dst.s_addr)
            return 0;
        if (src.s_addr != dst_ip.s_addr)
            return 0;
        if (memcmp(p+ah->ar_hln+4, ((struct sockaddr_ll *)&me)->sll_addr, ah->ar_hln))
            return 0;
    } else {
        /* DAD packet was:
           src_ip = 0 (or some src)
           src_hw = ME
           dst_ip = tested address
           dst_hw = <unspec>

        We fail, if receive request/reply with:
           src_ip = tested_address
           src_hw != ME
           if src_ip in request was not zero, check

```

```

        also that it matches to dst_ip, otherwise
        dst_ip/dst_hw do not matter.
    */
    if (src_ip.s_addr != dst.s_addr)
    return 0;
    if (memcmp(p, ((struct sockaddr_ll *)&me)->sll_addr, ((struct sockaddr_ll *)&me)->sll_halen) == 0)
    return 0;
    if (src.s_addr && src.s_addr != dst_ip.s_addr)
    return 0;
    }
    if (!quiet) {
    int s_printed = 0;
    printf("%s ", FROM->sll_pkttype==PACKET_HOST ? "Unicast" : "Broadcast");
    printf("%s from ", ah->ar_op == htons(ARPOP_REPLY) ? "reply" : "request");
    printf("%s [", inet_ntoa(src_ip));
    print_hex(p, ah->ar_hln);
    printf("] ");
    if (dst_ip.s_addr != src.s_addr) {
    printf("for %s ", inet_ntoa(dst_ip));
    s_printed = 1;
    }
    if (memcmp(p+ah->ar_hln+4, ((struct sockaddr_ll *)&me)->sll_addr, ah->ar_hln)) {
    if (!s_printed)
    printf("for ");
    printf("[");
    print_hex(p+ah->ar_hln+4, ah->ar_hln);
    printf("]");
    }
    if (last.tv_sec) {
    long usecs = (tv.tv_sec-last.tv_sec) * 1000000 +
    tv.tv_usec-last.tv_usec;
    long msecs = (usecs+500)/1000;
    usecs -= msecs*1000 - 500;
    printf(" %ld.%03ldms\n", msecs, usecs);
    } else {
    printf(" UNSOLICITED?\n");
    }
    fflush(stdout);
    }
    received++;
    if (FROM->sll_pkttype != PACKET_HOST)
    brd_rcv++;
    if (ah->ar_op == htons(ARPOP_REQUEST))
    req_rcv++;
    if (quit_on_reply || (count == 0 && received == sent))
    finish();
    if (!broadcast_only) {
    memcpy(((struct sockaddr_ll *)&he)->sll_addr, p, ((struct sockaddr_ll *)&me)->sll_halen);
    unicasting=1;
    }
    return 1;
    }

#ifdef USE_SYSFS

```

```

union sysfs_devattr_value {
    unsigned longulong;
    void*ptr;
};

enum {
    SYSFS_DEVATTR_IFINDEX,
    SYSFS_DEVATTR_FLAGS,
    SYSFS_DEVATTR_ADDR_LEN,
    #if 0
    SYSFS_DEVATTR_TYPE,
    SYSFS_DEVATTR_ADDRESS,
    #endif
    SYSFS_DEVATTR_BROADCAST,
    SYSFS_DEVATTR_NUM
};

struct sysfs_devattr_values
{
    char *ifname;
    union sysfs_devattr_valuevalue[SYSFS_DEVATTR_NUM];
};

static int sysfs_devattr_ulong_dec(char *ptr, struct sysfs_devattr_values *v, unsigned idx);
static int sysfs_devattr_ulong_hex(char *ptr, struct sysfs_devattr_values *v, unsigned idx);
static int sysfs_devattr_macaddr(char *ptr, struct sysfs_devattr_values *v, unsigned idx);

struct sysfs_devattrs {
    const char *name;
    int (*handler)(char *ptr, struct sysfs_devattr_values *v, unsigned int idx);
    int free;
} sysfs_devattrs[SYSFS_DEVATTR_NUM] = {
    [SYSFS_DEVATTR_IFINDEX] = {
        .name= "ifindex",
        .handler= sysfs_devattr_ulong_dec,
    },
    [SYSFS_DEVATTR_ADDR_LEN] = {
        .name= "addr_len",
        .handler= sysfs_devattr_ulong_dec,
    },
    [SYSFS_DEVATTR_FLAGS] = {
        .name= "flags",
        .handler= sysfs_devattr_ulong_hex,
    },
    #if 0
    [SYSFS_DEVATTR_TYPE] = {
        .name= "type",
        .handler= sysfs_devattr_ulong_dec,
    },
    [SYSFS_DEVATTR_ADDRESS] = {
        .name= "address",
        .handler= sysfs_devattr_macaddr,
        .free= 1,
    },
    #endif
};

```

```

[SYSFS_DEVATTR_BROADCAST] = {
.name= "broadcast",
.handler= sysfs_devattr_macaddr,
.free= 1,
},
};
#endif

/*
 * find_device()
 *
 * This function checks 1) if the device (if given) is okay for ARP,
 * or 2) find fist appropriate device on the system.
 *
 * Return value:
 * >0: Succeeded, and appropriate device not found.
 *     device.ifindex remains 0.
 * 0: Succeeded, and appropriate device found.
 *     device.ifindex is set.
 * <0: Failed. Support not found, or other
 *     system error. Try other method.
 *
 * If an appropriate device found, it is recorded inside the
 * "device" variable for later reference.
 *
 * We have several implementations for this.
 * by_ifaddrs():requires getifaddr() in glibc, and rtnetlink in
 * kernel. default and recommended for recent systems.
 * by_sysfs():requires libsysfs , and sysfs in kernel.
 * by_ioctl():unable to list devices without ipv4 address; this
 * means, you need to supply the device name for
 * DAD purpose.
 */
/* Common check for ifa->ifa_flags */
static int check_ifflags(unsigned int ifflags, int fatal)
{
    if (!(ifflags & IFF_UP)) {
        if (fatal) {
            if (!quiet)
                printf("Interface \"%s\" is down\n", device.name);
            exit(2);
        }
        return -1;
    }
    if (ifflags & (IFF_NOARP | IFF_LOOPBACK)) {
        if (fatal) {
            if (!quiet)
                printf("Interface \"%s\" is not ARPable\n", device.name);
            exit(dad ? 0 : 2);
        }
        return -1;
    }
    return 0;
}

```

```

static int find_device_by_ifaddrs(void)
{
#ifdef WITHOUT_IFADDRS
int rc;
struct ifaddrs *ifa0, *ifa;
int count = 0;

rc = getifaddrs(&ifa0);
if (rc) {
perror("getifaddrs");
return -1;
}

for (ifa = ifa0; ifa; ifa = ifa->ifa_next) {
if (!ifa->ifa_addr)
continue;
if (ifa->ifa_addr->sa_family != AF_PACKET)
continue;
if (device.name && ifa->ifa_name && strcmp(ifa->ifa_name, device.name))
continue;

if (check_ifflags(ifa->ifa_flags, device.name != NULL) < 0)
continue;

if (!((struct sockaddr_ll *)ifa->ifa_addr)->sll_halen)
continue;
if (!ifa->ifa_broadaddr)
continue;

device.ifa = ifa;

if (count++)
break;
}

if (count == 1 && device.ifa) {
device.ifindex = if_nametoindex(device.ifa->ifa_name);
if (!device.ifindex) {
perror("arping: if_nametoindex");
freeifaddrs(ifa0);
return -1;
}
device.name = device.ifa->ifa_name;
return 0;
}
return 1;
#else
return -1;
#endif
}

#ifdef USE_SYSFS
static void sysfs_devattr_values_init(struct sysfs_devattr_values *v, int do_free)
{

```

```
int i;
if (do_free) {
    free(v->ifname);
    for (i = 0; i < SYSFS_DEVATTR_NUM; i++) {
        if (sysfs_devattr[i].free)
            free(v->value[i].ptr);
    }
}
memset(v, 0, sizeof(*v));
}

static int sysfs_devattr_ulong(char *ptr, struct sysfs_devattr_values *v, unsigned int idx,
    unsigned int base)
{
    unsigned long *p;
    char *ep;

    if (!ptr || !v)
        return -1;

    p = &v->value[idx].ulong;
    errno = 0;
    *p = strtoul(ptr, &ep, base);
    if ((*ptr && isspace(*ptr & 0xff)) || errno || (*ep != '\0' && *ep != '\n'))
        goto out;

    return 0;
out:
    return -1;
}

static int sysfs_devattr_ulong_dec(char *ptr, struct sysfs_devattr_values *v, unsigned int idx)
{
    {
        int rc = sysfs_devattr_ulong(ptr, v, idx, 10);
        return rc;
    }
}

static int sysfs_devattr_ulong_hex(char *ptr, struct sysfs_devattr_values *v, unsigned int idx)
{
    {
        int rc = sysfs_devattr_ulong(ptr, v, idx, 16);
        return rc;
    }
}

static int sysfs_devattr_macaddr(char *ptr, struct sysfs_devattr_values *v, unsigned int idx)
{
    {
        unsigned char *m;
        int i;
        unsigned int addrlen;

        if (!ptr || !v)
            return -1;

        addrlen = v->value[SYSFS_DEVATTR_ADDR_LEN].ulong;
        m = malloc(addrlen);
```

```

    for (i = 0; i < addrlen; i++) {
        if (i && *(ptr + i * 3 - 1) != ':')
            goto out;
        if (sscanf(ptr + i * 3, "%02hhx", &m[i]) != 1)
            goto out;
    }

    v->value[idx].ptr = m;
    return 0;
out:
    free(m);
    return -1;
}
#endif

int find_device_by_sysfs(void)
{
    int rc = -1;
#ifdef USE_SYSFS
    struct sysfs_class *cls_net;
    struct dlist *dev_list;
    struct sysfs_class_device *dev;
    struct sysfs_attribute *dev_attr;
    struct sysfs_devattr_values sysfs_devattr_values;
    int count = 0;

    if (!device.sysfs) {
        device.sysfs = malloc(sizeof(*device.sysfs));
        sysfs_devattr_values_init(device.sysfs, 0);
    }

    cls_net = sysfs_open_class("net");
    if (!cls_net) {
        perror("sysfs_open_class");
        return -1;
    }

    dev_list = sysfs_get_class_devices(cls_net);
    if (!dev_list) {
        perror("sysfs_get_class_devices");
        goto out;
    }

    sysfs_devattr_values_init(&sysfs_devattr_values, 0);

    dlist_for_each_data(dev_list, dev, struct sysfs_class_device) {
        int i;
        int rc = -1;

        if (device.name && strcmp(dev->name, device.name))
            goto do_next;

        sysfs_devattr_values_init(&sysfs_devattr_values, 1);

```



```

for (i = 0; i < SYSFS_DEVATTR_NUM; i++) {

    dev_attr = sysfs_get_classdev_attr(dev, sysfs_devattr[i].name);
    if (!dev_attr) {
        perror("sysfs_get_classdev_attr");
        rc = -1;
        break;
    }
    if (sysfs_read_attribute(dev_attr)) {
        perror("sysfs_read_attribute");
        rc = -1;
        break;
    }
    rc = sysfs_devattr[i].handler(dev_attr->value, &sysfs_devattr_values, i);

    if (rc < 0)
        break;
    }

    if (rc < 0)
        goto do_next;

    if (check_iflags(sysfs_devattr_values.value[SYSFS_DEVATTR_FLAGS].ulong,
        device.name != NULL) < 0)
        goto do_next;

    if (!sysfs_devattr_values.value[SYSFS_DEVATTR_ADDR_LEN].ulong)
        goto do_next;

    if (device.sysfs->value[SYSFS_DEVATTR_IFINDEX].ulong) {
        if (device.sysfs->value[SYSFS_DEVATTR_FLAGS].ulong & IFF_RUNNING)
            goto do_next;
    }

    sysfs_devattr_values.ifname = strdup(dev->name);
    if (!sysfs_devattr_values.ifname) {
        perror("malloc");
        goto out;
    }

    sysfs_devattr_values_init(device.sysfs, 1);
    memcpy(device.sysfs, &sysfs_devattr_values, sizeof(*device.sysfs));
    sysfs_devattr_values_init(&sysfs_devattr_values, 0);

    if (count++)
        break;

    continue;
do_next:
    sysfs_devattr_values_init(&sysfs_devattr_values, 1);
    }

    if (count == 1) {
        device.ifindex = device.sysfs->value[SYSFS_DEVATTR_IFINDEX].ulong;

```

```

device.name = device.sysfs->ifname;
}
rc = !device.ifindex;
out:
sysfs_close_class(cls_net);
#endif
return rc;
}

static int check_device_by_ioctl(int s, struct ifreq *ifr)
{
if (ioctl(s, SIOCGIFFLAGS, ifr) < 0) {
perror("ioctl(SIOCGIFFLAGS)");
return -1;
}

if (check_iflags(ifr->ifr_flags, device.name != NULL) < 0)
return 1;

if (ioctl(s, SIOCGIFINDEX, ifr) < 0) {
perror("ioctl(SIOCGIFINDEX)");
return -1;
}

return 0;
}

static int find_device_by_ioctl(void)
{
int s;
struct ifreq *ifr0, *ifr, *ifr_end;
size_t ifrsize = sizeof(*ifr);
struct ifconf ifc;
static struct ifreq ifrbuf;
int count = 0;

s = socket(AF_INET, SOCK_DGRAM, 0);
if (s < 0) {
perror("socket");
return -1;
}

memset(&ifrbuf, 0, sizeof(ifrbuf));

if (device.name) {
strncpy(ifrbuf.ifr_name, device.name, sizeof(ifrbuf.ifr_name) - 1);
if (check_device_by_ioctl(s, &ifrbuf))
goto out;
count++;
} else {
do {
int rc;
ifr0 = malloc(ifrsize);
if (!ifr0) {
perror("malloc");

```

```
goto out;
}

ifc.ifc_buf = (char *)ifr0;
ifc.ifc_len = ifrsize;

rc = ioctl(s, SIOCGIFCONF, &ifc);
if (rc < 0) {
perror("ioctl(SIOCGIFCONF");
goto out;
}

if (ifc.ifc_len + sizeof(*ifr0) + sizeof(struct sockaddr_storage) - sizeof(struct sockaddr) <= ifrsize)
break;
ifrsize *= 2;
free(ifr0);
ifr0 = NULL;
} while (ifrsize < INT_MAX / 2);

if (!ifr0) {
fprintf(stderr, "arping: too many interfaces!?\n");
goto out;
}

ifr_end = (struct ifreq *)(((char *)ifr0) + ifc.ifc_len - sizeof(*ifr0));
for (ifr = ifr0; ifr <= ifr_end; ifr++) {
if (check_device_by_ioctl(s, &ifrbuf))
continue;
memcpy(&ifrbuf.ifr_name, ifr->ifr_name, sizeof(ifrbuf.ifr_name));
if (count++)
break;
}
}

close(s);

if (count == 1) {
device.ifindex = ifrbuf.ifr_ifindex;
device.name = ifrbuf.ifr_name;
}
return !device.ifindex;
out:
close(s);
return -1;
}

static int find_device(void)
{
int rc;
rc = find_device_by_ifaddrs();
if (rc >= 0)
goto out;
rc = find_device_by_sysfs();
if (rc >= 0)
```

```

goto out;
rc = find_device_by_ioctl();
out:
return rc;
}

/*
 * set_device_broadcast()
 *
 * This fills the device "broadcast address"
 * based on information found by find_device() funcion.
 */
static int set_device_broadcast_ifaddrs_one(struct device *device, unsigned char *ba, size_t balen, int fatal)
{
#ifdef WITHOUT_IFADDRS
struct ifaddrs *ifa;
struct sockaddr_ll *sll;

if (!device)
return -1;

ifa = device->ifa;
if (!ifa)
return -1;

sll = (struct sockaddr_ll *)ifa->ifa_broadaddr;

if (sll->sll_halen != balen) {
if (fatal) {
if (!quiet)
printf("Address length does not match...\n");
exit(2);
}
return -1;
}
memcpy(ba, sll->sll_addr, sll->sll_halen);
return 0;
#else
return -1;
#endif
}

int set_device_broadcast_sysfs(struct device *device, unsigned char *ba, size_t balen)
{
#ifdef USE_SYSFS
struct sysfs_devattr_values *v;
if (!device)
return -1;
v = device->sysfs;
if (!v)
return -1;
if (v->value[SYSFS_DEVATTR_ADDR_LEN].ulong != balen)
return -1;
memcpy(ba, v->value[SYSFS_DEVATTR_BROADCAST].ptr, balen);
return 0;
#else

```

```
return -1;
#endif
}

static int set_device_broadcast_fallback(struct device *device, unsigned char *ba, size_t balen)
{
    if (!quiet)
        fprintf(stderr, "WARNING: using default broadcast address.\n");
    memset(ba, -1, balen);
    return 0;
}

static void set_device_broadcast(struct device *dev, unsigned char *ba, size_t balen)
{
    if (!set_device_broadcast_ifaddrs_one(dev, ba, balen, 0))
        return;
    if (!set_device_broadcast_sysfs(dev, ba, balen))
        return;
    set_device_broadcast_fallback(dev, ba, balen);
}

int
main(int argc, char **argv)
{
    int socket_errno;
    int ch;

    limit_capabilities();

#ifdef USE_IDN
    setlocale(LC_ALL, "");
#endif

    enable_capability_raw();

    s = socket(PF_PACKET, SOCK_DGRAM, 0);
    socket_errno = errno;

    disable_capability_raw();

    while ((ch = getopt(argc, argv, "h?bfDUAqc:w:s:I:V")) != EOF) {
        switch(ch) {
            case 'b':
                broadcast_only=1;
                break;
            case 'D':
                dad++;
                quit_on_reply=1;
                break;
            case 'U':
                unsolicited++;
                break;
            case 'A':
                advert++;
                break;
        }
    }
}
```

```

    unsolicited++;
    break;
    case 'q':
    quiet++;
    break;
    case 'c':
    count = atoi(optarg);
    break;
    case 'w':
    timeout = atoi(optarg);
    break;
    case 'I':
    device.name = optarg;
    break;
    case 'f':
    quit_on_reply=1;
    break;
    case 's':
    source = optarg;
    break;
    case 'V':
    printf("arping utility, iputils-%s\n", SNAPSHOT);
    exit(0);
    case 'h':
    case '?':
    default:
    usage();
    }
    }
    argc -= optind;
    argv += optind;

    if (argc != 1)
    usage();

    target = *argv;

    if (device.name && !*device.name)
    device.name = NULL;

    if (s < 0) {
    errno = socket_errno;
    perror("arping: socket");
    exit(2);
    }

    if (find_device() < 0)
    exit(2);

    if (!device.ifindex) {
    if (device.name) {
    fprintf(stderr, "arping: Device %s not available.\n", device.name);
    exit(2);
    }
    fprintf(stderr, "arping: device (option -I) is required.\n");

```

```
usage();
}

if (inet_aton(target, &dst) != 1) {
    struct hostent *hp;
    char *idn = target;
#ifdef USE_IDN
    int rc;

    rc = idna_to_ascii_lz(target, &idn, 0);

    if (rc != IDNA_SUCCESS) {
        fprintf(stderr, "arping: IDN encoding failed: %s\n", idna_strerror(rc));
        exit(2);
    }
#endif

    hp = gethostbyname2(idn, AF_INET);
    if (!hp) {
        fprintf(stderr, "arping: unknown host %s\n", target);
        exit(2);
    }

#ifdef USE_IDN
    free(idn);
#endif

    memcpy(&dst, hp->h_addr, 4);
}

if (source && inet_aton(source, &src) != 1) {
    fprintf(stderr, "arping: invalid source %s\n", source);
    exit(2);
}

if (!dad && unsolicited && src.s_addr == 0)
    src = dst;

if (!dad || src.s_addr) {
    struct sockaddr_in saddr;
    int probe_fd = socket(AF_INET, SOCK_DGRAM, 0);

    if (probe_fd < 0) {
        perror("socket");
        exit(2);
    }
    if (device.name) {
        enable_capability_raw();

        if (setsockopt(probe_fd, SOL_SOCKET, SO_BINDTODEVICE, device.name, strlen(device.name)+1) == -1)
            perror("WARNING: interface is ignored");

        disable_capability_raw();
    }
}
```

```

memset(&saddr, 0, sizeof(saddr));
saddr.sin_family = AF_INET;
if (src.s_addr) {
saddr.sin_addr = src;
if (bind(probe_fd, (struct sockaddr*)&saddr, sizeof(saddr)) == -1) {
perror("bind");
exit(2);
}
} else if (!dad) {
int on = 1;
socklen_t alen = sizeof(saddr);

saddr.sin_port = htons(1025);
saddr.sin_addr = dst;

if (setsockopt(probe_fd, SOL_SOCKET, SO_DONTROUTE, (char*)&on, sizeof(on)) == -1)
perror("WARNING: setsockopt(SO_DONTROUTE)");
if (connect(probe_fd, (struct sockaddr*)&saddr, sizeof(saddr)) == -1) {
perror("connect");
exit(2);
}
if (getsockname(probe_fd, (struct sockaddr*)&saddr, &alen) == -1) {
perror("getsockname");
exit(2);
}
src = saddr.sin_addr;
}
close(probe_fd);
};

((struct sockaddr_ll *)&me)->sll_family = AF_PACKET;
((struct sockaddr_ll *)&me)->sll_ifindex = device.ifindex;
((struct sockaddr_ll *)&me)->sll_protocol = htons(ETH_P_ARP);
if (bind(s, (struct sockaddr*)&me, sizeof(me)) == -1) {
perror("bind");
exit(2);
}

if (1) {
socklen_t alen = sizeof(me);
if (getsockname(s, (struct sockaddr*)&me, &alen) == -1) {
perror("getsockname");
exit(2);
}
}
if (((struct sockaddr_ll *)&me)->sll_halen == 0) {
if (!quiet)
printf("Interface \"%s\" is not ARPable (no ll address)\n", device.name);
exit(dad?0:2);
}

he = me;

set_device_broadcast(&device, ((struct sockaddr_ll *)&he)->sll_addr,
((struct sockaddr_ll *)&he)->sll_halen);

```



```

if (!quiet) {
printf("ARPING %s ", inet_ntoa(dst));
printf("from %s %s\n", inet_ntoa(src), device.name ? : "");
}

if (!src.s_addr && !dad) {
fprintf(stderr, "arping: no source address in not-DAD mode\n");
exit(2);
}

drop_capabilities();

set_signal(SIGINT, finish);
set_signal(SIGALRM, catcher);

catcher();

while(1) {
sigset_t sset, osset;
unsigned char packet[4096];
struct sockaddr_storage from;
socklen_t alen = sizeof(from);
int cc;

sigemptyset(&sset);
sigaddset(&sset, SIGALRM);
sigaddset(&sset, SIGINT);
/* Unblock SIGALRM so that the previously called alarm()
 * can prevent recvfrom from blocking forever in case the
 * inherited procmask is blocking SIGALRM and no packet
 * is received. */
sigprocmask(SIG_UNBLOCK, &sset, &osset);

if ((cc = recvfrom(s, packet, sizeof(packet), 0,
    (struct sockaddr *)&from, &alen)) < 0) {
perror("arping: recvfrom");
continue;
}

sigprocmask(SIG_BLOCK, &sset, NULL);
recv_pack(packet, cc, (struct sockaddr_ll *)&from);
sigprocmask(SIG_SETMASK, &osset, NULL);
}
}

```

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright

- notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
 3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to

know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather,

the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision` (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
/*
 * Copyright (c) 1989 The Regents of the University of California.
 * All rights reserved.
 *
 * This code is derived from software contributed to Berkeley by
```

```

* Mike Muuss.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
*   must display the following acknowledgement:
* This product includes software developed by the University of
* California, Berkeley and its contributors.
* 4. Neither the name of the University nor the names of its contributors
*   may be used to endorse or promote products derived from this software
*   without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*/

#ifdef lint
char copyright[] =
"@(#) Copyright (c) 1989 The Regents of the University of California.\n\
All rights reserved.\n";
#endif /* not lint */

/*
*PING.C
*
* Using the InterNet Control Message Protocol (ICMP) "ECHO" facility,
* measure round-trip-delays and packet loss across network paths.
*
* Author -
*Mike Muuss
*U. S. Army Ballistic Research Laboratory
*December, 1983
*
* Status -
*Public Domain. Distribution Unlimited.
* Bugs -
*More statistics could always be gathered.
*If kernel does not support non-raw ICMP sockets,
*this program has to run SUID to ROOT or with

```



```

*net_cap_raw enabled.
*/

#include "ping_common.h"

#include <netinet/ip.h>
#include <netinet/ip_icmp.h>
#ifndef WITHOUT_IFADDRS
#include <ifaddrs.h>
#endif

#ifndef ICMP_FILTER
#define ICMP_FILTER1
struct icmp_filter {
    __u32 data;
};
#endif

#define MAXIPLLEN60
#define MAXICMPLEN76
#define NROUTES9 /* number of record route slots */
#define TOS_MAX255 /* 8-bit TOS field */
#define MAX_HOSTNAMELENNI_MAXHOST

static int ts_type;
static int nroute = 0;
static __u32 route[10];

struct sockaddr_in where_to; /* who to ping */
int optlen = 0;
int settos = 0; /* Set TOS, Precedence or other QOS options */
int icmp_sock; /* socket file descriptor */
extern int using_ping_socket;
u_char outpack[0x10000];
int maxpacket = sizeof(outpack);

static int broadcast_pings = 0;

static char *pr_addr(__u32);
static void pr_options(unsigned char *cp, int hlen);
static void pr_iph(struct iphdr *ip);
static void usage(void) __attribute__((noreturn));
static u_short in_cksum(const u_short *addr, int len, u_short salt);
static void pr_icmph(__u8 type, __u8 code, __u32 info, struct icmphdr *icp);
static int parsetos(char *str);

static struct {
    struct cmsghdr cm;
    struct in_pktinfo ipi;
} cmsg = { {sizeof(struct cmsghdr) + sizeof(struct in_pktinfo), SOL_IP, IP_PKTINFO},
    {0,}};

```



```
int cmsg_len;

struct sockaddr_in source;
char *device;
int pmtudisc = -1;

int
main(int argc, char **argv)
{
    struct hostent *hp;
    int ch, hold, packlen;
    int socket_errno;
    u_char *packet;
    char *target;
#ifdef USE_IDN
    char *hnamebuf = NULL;
#else
    char hnamebuf[MAX_HOSTNAMELEN];
#endif
    char rspace[3 + 4 * NROUTES + 1]; /* record route space */

    limit_capabilities();

#ifdef USE_IDN
    setlocale(LC_ALL, "");
#endif

    enable_capability_raw();
    icmp_sock = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP);
    disable_capability_raw();

    if (icmp_sock < 0) {
        icmp_sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_ICMP);
        using_ping_socket = 1;
        working_recverr = 1;
    }
    socket_errno = errno;

    source.sin_family = AF_INET;

    preload = 1;
    while ((ch = getopt(argc, argv, COMMON_OPTSTR "bRT:")) != EOF) {
        switch(ch) {
            case 'b':
                broadcast_pings = 1;
                break;
            case 'Q':
                settos = parsetos(optarg);
                if (settos &&
                    (setsockopt(icmp_sock, IPPROTO_IP, IP_TOS,
                        (char *)&settos, sizeof(int)) < 0)) {
                    perror("ping: error setting QOS sockopts");
                }
            }
        }
    }
}
```

```

    exit(2);
}
break;
case 'R':
if (options & F_TIMESTAMP) {
fprintf(stderr, "Only one of -T or -R may be used\n");
exit(2);
}
options |= F_RROUTE;
break;
case 'T':
if (options & F_RROUTE) {
fprintf(stderr, "Only one of -T or -R may be used\n");
exit(2);
}
options |= F_TIMESTAMP;
if (strcmp(optarg, "tsonly") == 0)
ts_type = IPOPT_TS_TSONLY;
else if (strcmp(optarg, "tsandaddr") == 0)
ts_type = IPOPT_TS_TSANDADDR;
else if (strcmp(optarg, "tsprespec") == 0)
ts_type = IPOPT_TS_PRESPEC;
else {
fprintf(stderr, "Invalid timestamp type\n");
exit(2);
}
break;
case 'I':
{
#ifdef 0
char dummy;
int i1, i2, i3, i4;

if (sscanf(optarg, "%u.%u.%u.%u%c",
&i1, &i2, &i3, &i4, &dummy) == 4) {
__u8 *ptr;
ptr = (__u8*)&source.sin_addr;
ptr[0] = i1;
ptr[1] = i2;
ptr[2] = i3;
ptr[3] = i4;
options |= F_STRICTSOURCE;
} else {
device = optarg;
}
#else
if (inet_pton(AF_INET, optarg, &source.sin_addr) > 0)
options |= F_STRICTSOURCE;
else
device = optarg;
#endif
break;
}
case 'M':
if (strcmp(optarg, "do") == 0)

```

```

pmtudisc = IP_PMTUDISC_DO;
else if (strcmp(optarg, "dont") == 0)
pmtudisc = IP_PMTUDISC_DONT;
else if (strcmp(optarg, "want") == 0)
pmtudisc = IP_PMTUDISC_WANT;
else {
fprintf(stderr, "ping: wrong value for -M: do, dont, want are valid ones.\n");
exit(2);
}
break;
case 'V':
printf("ping utility, iputils-%s\n", SNAPSHOT);
exit(0);
COMMON_OPTIONS
common_options(ch);
break;
default:
usage();
}
}
argc -= optind;
argv += optind;

if (argc == 0)
usage();
if (argc > 1) {
if (options & F_RROUTE)
usage();
else if (options & F_TIMESTAMP) {
if (ts_type != IPOPT_TS_PRESPEC)
usage();
}
if (argc > 5)
usage();
} else {
if (argc > 10)
usage();
options |= F_SOURCEROUTE;
}
}
while (argc > 0) {
target = *argv;

memset((char *)&whereto, 0, sizeof(whereto));
whereto.sin_family = AF_INET;
if (inet_aton(target, &whereto.sin_addr) == 1) {
hostname = target;
if (argc == 1)
options |= F_NUMERIC;
} else {
char *idn;
#ifdef USE_IDN
int rc;

if (hnamebuf) {

```

```

    free(hnamebuf);
    hnamebuf = NULL;
}

rc = idna_to_ascii_lz(target, &idn, 0);
if (rc != IDNA_SUCCESS) {
    fprintf(stderr, "ping: IDN encoding failed: %s\n", idna_strerror(rc));
    exit(2);
}
#else
idn = target;
#endif
hp = gethostbyname2(idn, AF_INET);
if (!hp) {
    fprintf(stderr, "ping: unknown host %s\n", target);
    exit(2);
}
#ifdef USE_IDN
free(idn);
#endif
memcpy(&where.to.sin_addr, hp->h_addr, 4);
#ifdef USE_IDN
if (idna_to_unicode_lzlz(hp->h_name, &hnamebuf, 0) != IDNA_SUCCESS) {
    hnamebuf = strdup(hp->h_name);
    if (!hnamebuf) {
        perror("ping: strdup");
        exit(-1);
    }
}
#else
strncpy(hnamebuf, hp->h_name, sizeof(hnamebuf) - 1);
hnamebuf[sizeof(hnamebuf) - 1] = 0;
#endif
hostname = hnamebuf;
}
if (argc > 1)
    route[nroute++] = where.to.sin_addr.s_addr;
argc--;
argv++;
}

if (source.sin_addr.s_addr == 0) {
    socklen_t alen;
    struct sockaddr_in dst = where.to;
    int probe_fd = socket(AF_INET, SOCK_DGRAM, 0);

    if (probe_fd < 0) {
        perror("socket");
        exit(2);
    }
    if (device) {
        struct ifreq ifr;
        int rc;

        memset(&ifr, 0, sizeof(ifr));

```

```

strncpy(ifr.ifr_name, device, IFNAMSIZ-1);

enable_capability_raw();
rc = setsockopt(probe_fd, SOL_SOCKET, SO_BINDTODEVICE, device, strlen(device)+1);
disable_capability_raw();

if (rc == -1) {
if (IN_MULTICAST(ntohl(dst.sin_addr.s_addr))) {
struct ip_mreqn imr;
if (ioctl(probe_fd, SIOCGIFINDEX, &ifr) < 0) {
fprintf(stderr, "ping: unknown iface %s\n", device);
exit(2);
}
memset(&imr, 0, sizeof(imr));
imr.imr_ifindex = ifr.ifr_ifindex;
if (setsockopt(probe_fd, SOL_IP, IP_MULTICAST_IF, &imr, sizeof(imr)) == -1) {
perror("ping: IP_MULTICAST_IF");
exit(2);
}
} else {
perror("ping: SO_BINDTODEVICE");
exit(2);
}
}

if (settos &&
    setsockopt(probe_fd, IPPROTO_IP, IP_TOS, (char *)&settos, sizeof(int)) < 0)
perror("Warning: error setting QoS sockopts");

dst.sin_port = htons(1025);
if (nroute)
dst.sin_addr.s_addr = route[0];
if (connect(probe_fd, (struct sockaddr*)&dst, sizeof(dst)) == -1) {
if (errno == EACCES) {
if (broadcast_pings == 0) {
fprintf(stderr, "Do you want to ping broadcast? Then -b\n");
exit(2);
}
fprintf(stderr, "WARNING: pinging broadcast address\n");
if (setsockopt(probe_fd, SOL_SOCKET, SO_BROADCAST,
    &broadcast_pings, sizeof(broadcast_pings)) < 0) {
perror("can't set broadcasting");
exit(2);
}
}
if (connect(probe_fd, (struct sockaddr*)&dst, sizeof(dst)) == -1) {
perror("connect");
exit(2);
}
} else {
perror("connect");
exit(2);
}
}
}

```

```

alen = sizeof(source);
if (getsockname(probe_fd, (struct sockaddr*)&source, &alen) == -1) {
    perror("getsockname");
    exit(2);
}
source.sin_port = 0;

#ifdef WITHOUT_IFADDRS
if (device) {
    struct ifaddrs *ifa0, *ifa;
    int ret;

    ret = getifaddrs(&ifa0);
    if (ret) {
        fprintf(stderr, "getifaddrs() failed.\n");
        exit(2);
    }
    for (ifa = ifa0; ifa; ifa = ifa->ifa_next) {
        if (!ifa->ifa_addr || ifa->ifa_addr->sa_family != AF_INET)
            continue;
        if (!strcmp(ifa->ifa_name, device, sizeof(device) - 1) &&
            !memcmp(&((struct sockaddr_in *)ifa->ifa_addr->sin_addr,
                &source.sin_addr, sizeof(source.sin_addr))))
            break;
    }
    freeifaddrs(ifa0);
    if (!ifa)
        fprintf(stderr, "ping: Warning: source address might be selected on device other than %s.\n", device);
    }
#endif
close(probe_fd);
} while (0);

if (whereto.sin_addr.s_addr == 0)
    whereto.sin_addr.s_addr = source.sin_addr.s_addr;

if (icmp_sock < 0) {
    errno = socket_errno;
    perror("ping: icmp open socket");
    exit(2);
}

if (device) {
    struct ifreq ifr;

    memset(&ifr, 0, sizeof(ifr));
    strncpy(ifr.ifr_name, device, IFNAMSIZ-1);
    if (ioctl(icmp_sock, SIOCGIFINDEX, &ifr) < 0) {
        fprintf(stderr, "ping: unknown iface %s\n", device);
        exit(2);
    }
    cmsg.ipi.ipi_ifindex = ifr.ifr_ifindex;
    cmsg_len = sizeof(cmsg);
}

```

```

if (broadcast_pings || IN_MULTICAST(ntohl(wheretos.sin_addr.s_addr))) {
if (uid) {
if (interval < 1000) {
fprintf(stderr, "ping: broadcast ping with too short interval.\n");
exit(2);
}
if (pmtudisc >= 0 && pmtudisc != IP_PMTUDISC_DO) {
fprintf(stderr, "ping: broadcast ping does not fragment.\n");
exit(2);
}
}
if (pmtudisc < 0)
pmtudisc = IP_PMTUDISC_DO;
}

if (pmtudisc >= 0) {
if (setsockopt(icmp_sock, SOL_IP, IP_MTU_DISCOVER, &pmtudisc, sizeof(pmtudisc)) == -1) {
perror("ping: IP_MTU_DISCOVER");
exit(2);
}
}

if ((options & F_STRICTSOURCE) &&
    bind(icmp_sock, (struct sockaddr*)&source, sizeof(source)) == -1) {
perror("bind");
exit(2);
}

if (!using_ping_socket) {
struct icmp_filter filt;
filt.data = ~(1<<ICMP_SOURCE_QUENCH|
              1<<ICMP_DEST_UNREACH|
              1<<ICMP_TIME_EXCEEDED|
              1<<ICMP_PARAMETERPROB|
              1<<ICMP_REDIRECT|
              1<<ICMP_ECHOREPLY);
if (setsockopt(icmp_sock, SOL_RAW, ICMP_FILTER, (char*)&filt, sizeof(filt)) == -1)
perror("WARNING: setsockopt(ICMP_FILTER)");
}

hold = 1;
if (setsockopt(icmp_sock, SOL_IP, IP_RECVERR, (char *)&hold, sizeof(hold)))
fprintf(stderr, "WARNING: your kernel is veeery old. No problems.\n");
if (using_ping_socket) {
if (setsockopt(icmp_sock, SOL_IP, IP_RECVTTL,
              (char *)&hold, sizeof(hold)))
perror("WARNING: setsockopt(IP_RECVTTL)");
if (setsockopt(icmp_sock, SOL_IP, IP_RETOPTS,
              (char *)&hold, sizeof(hold)))
perror("WARNING: setsockopt(IP_RETOPTS)");
}

/* record route option */
if (options & F_RROUTE) {

```

```

memset(rspace, 0, sizeof(rspace));
rspace[0] = IPOPT_NOP;
rspace[1+IPOPT_OPTVAL] = IPOPT_RR;
rspace[1+IPOPT_OLEN] = sizeof(rspace)-1;
rspace[1+IPOPT_OFFSET] = IPOPT_MINOFF;
optlen = 40;
if (setsockopt(icmp_sock, IPPROTO_IP, IP_OPTIONS, rspace, sizeof(rspace)) < 0) {
perror("ping: record route");
exit(2);
}
}
if (options & F_TIMESTAMP) {
memset(rspace, 0, sizeof(rspace));
rspace[0] = IPOPT_TIMESTAMP;
rspace[1] = (ts_type==IPOPT_TS_TSONLY ? 40 : 36);
rspace[2] = 5;
rspace[3] = ts_type;
if (ts_type == IPOPT_TS_PRESPEC) {
int i;
rspace[1] = 4+nroute*8;
for (i=0; i<nroute; i++)
*(__u32*)&rspace[4+i*8] = route[i];
}
if (setsockopt(icmp_sock, IPPROTO_IP, IP_OPTIONS, rspace, rspace[1]) < 0) {
rspace[3] = 2;
if (setsockopt(icmp_sock, IPPROTO_IP, IP_OPTIONS, rspace, rspace[1]) < 0) {
perror("ping: ts option");
exit(2);
}
}
optlen = 40;
}
if (options & F_SOURCEROUTE) {
int i;
memset(rspace, 0, sizeof(rspace));
rspace[0] = IPOPT_NOOP;
rspace[1+IPOPT_OPTVAL] = (options & F_SO_DONTROUTE) ? IPOPT_SSRR
: IPOPT_LSRR;
rspace[1+IPOPT_OLEN] = 3 + nroute*4;
rspace[1+IPOPT_OFFSET] = IPOPT_MINOFF;
for (i=0; i<nroute; i++)
*(__u32*)&rspace[4+i*4] = route[i];

if (setsockopt(icmp_sock, IPPROTO_IP, IP_OPTIONS, rspace, 4 + nroute*4) < 0) {
perror("ping: record route");
exit(2);
}
optlen = 40;
}

/* Estimate memory eaten by single packet. It is rough estimate.
 * Actually, for small datalen's it depends on kernel side a lot. */
hold = datalen + 8;
hold += ((hold+511)/512)*(optlen + 20 + 16 + 64 + 160);
sock_setbufs(icmp_sock, hold);

```



```

if (broadcast_pings) {
if (setsockopt(icmp_sock, SOL_SOCKET, SO_BROADCAST,
    &broadcast_pings, sizeof(broadcast_pings)) < 0) {
perror ("ping: can't set broadcasting");
exit(2);
}
}

if (options & F_NOLOOP) {
int loop = 0;
if (setsockopt(icmp_sock, IPPROTO_IP, IP_MULTICAST_LOOP,
    &loop, 1) == -1) {
perror ("ping: can't disable multicast loopback");
exit(2);
}
}
if (options & F_TTL) {
int ittl = ttl;
if (setsockopt(icmp_sock, IPPROTO_IP, IP_MULTICAST_TTL,
    &ttl, 1) == -1) {
perror ("ping: can't set multicast time-to-live");
exit(2);
}
if (setsockopt(icmp_sock, IPPROTO_IP, IP_TTL,
    &ittl, sizeof(ittl)) == -1) {
perror ("ping: can't set unicast time-to-live");
exit(2);
}
}

if (datalen > 0xFFFF - 8 - optlen - 20) {
if (uid || datalen > sizeof(outpack)-8) {
fprintf(stderr, "Error: packet size %d is too large. Maximum is %d\n", datalen, 0xFFFF-8-20-optlen);
exit(2);
}
/* Allow small oversize to root yet. It will cause EMSGSIZE. */
fprintf(stderr, "WARNING: packet size %d is too large. Maximum is %d\n", datalen, 0xFFFF-8-20-optlen);
}

if (datalen >= sizeof(struct timeval))/* can we time transfer */
timing = 1;
packlen = datalen + MAXIPLen + MAXICMPLEN;
if (!(packet = (u_char *)malloc((u_int)packlen))) {
fprintf(stderr, "ping: out of memory.\n");
exit(2);
}

printf("PING %s (%s) ", hostname, inet_ntoa(whereto.sin_addr));
if (device || (options & F_STRICTSOURCE))
printf("from %s %s: ", inet_ntoa(source.sin_addr), device ? : "");
printf("%d(%d) bytes of data.\n", datalen, datalen+8+optlen+20);

setup(icmp_sock);

```

```

main_loop(icmp_sock, packet, packlen);
}

int receive_error_msg()
{
int res;
char cbuf[512];
struct iovec iov;
struct msghdr msg;
struct cmsghdr *cmsg;
struct sock_extended_err *e;
struct icmphdr icmph;
struct sockaddr_in target;
int net_errors = 0;
int local_errors = 0;
int saved_errno = errno;

iov.iov_base = &icmph;
iov.iov_len = sizeof(icmph);
msg.msg_name = (void*)&target;
msg.msg_namelen = sizeof(target);
msg.msg_iov = &iov;
msg.msg_iovlen = 1;
msg.msg_flags = 0;
msg.msg_control = cbuf;
msg.msg_controllen = sizeof(cbuf);

res = recvmsg(icmp_sock, &msg, MSG_ERRQUEUE|MSG_DONTWAIT);
if (res < 0)
goto out;

e = NULL;
for (cmsg = CMSG_FIRSTHDR(&msg); cmsg; cmsg = CMSG_NXTHDR(&msg, cmsg)) {
if (cmsg->cmsg_level == SOL_IP) {
if (cmsg->cmsg_type == IP_RECVERR)
e = (struct sock_extended_err *)CMSG_DATA(cmsg);
}
}
if (e == NULL)
abort();

if (e->ee_origin == SO_EE_ORIGIN_LOCAL) {
local_errors++;
if (options & F_QUIET)
goto out;
if (options & F_FLOOD)
write_stdout("E", 1);
else if (e->ee_errno != EMSGSIZE)
fprintf(stderr, "ping: local error: %s\n", strerror(e->ee_errno));
else
fprintf(stderr, "ping: local error: Message too long, mtu=%u\n", e->ee_info);
nerrors++;
} else if (e->ee_origin == SO_EE_ORIGIN_ICMP) {

```

```

struct sockaddr_in *sin = (struct sockaddr_in*)(e+1);

if (res < sizeof(icmp_h) ||
    target.sin_addr.s_addr != where.to.sin_addr.s_addr ||
    icmp_h.type != ICMP_ECHO ||
    !is_ours(icmp_h.un.echo.id)) {
/* Not our error, not an error at all. Clear. */
saved_errno = 0;
goto out;
}

acknowledge(ntohs(icmp_h.un.echo.sequence));

if (!using_ping_socket && !working_recverr) {
struct icmp_filter filt;
working_recverr = 1;
/* OK, it works. Add stronger filter. */
filt.data = ~(1<<ICMP_SOURCE_QUENCH|
               1<<ICMP_REDIRECT|
               1<<ICMP_ECHOREPLY);
if (setsockopt(icmp_sock, SOL_RAW, ICMP_FILTER, (char*)&filt, sizeof(filt)) == -1)
perror("\rWARNING: setsockopt(ICMP_FILTER)");
}

net_errors++;
nerrors++;
if (options & F_QUIET)
goto out;
if (options & F_FLOOD) {
write_stdout("\bE", 2);
} else {
print_timestamp();
printf("From %s icmp_seq=%u ", pr_addr(sin->sin_addr.s_addr), ntohs(icmp_h.un.echo.sequence));
pr_icmph(e->ee_type, e->ee_code, e->ee_info, NULL);
fflush(stdout);
}
}

out:
errno = saved_errno;
return net_errors ? : -local_errors;
}

/*
 * pinger --
 * Compose and transmit an ICMP ECHO REQUEST packet. The IP packet
 * will be added on by the kernel. The ID field is our UNIX process ID,
 * and the sequence number is an ascending integer. The first 8 bytes
 * of the data portion are used to hold a UNIX "timeval" struct in VAX
 * byte-order, to compute the round-trip time.
 */
int send_probe()
{
struct icmp_hdr *icp;

```

```

int cc;
int i;

icp = (struct icmphdr *)outpack;
icp->type = ICMP_ECHO;
icp->code = 0;
icp->checksum = 0;
icp->un.echo.sequence = htons(ntransmitted+1);
icp->un.echo.id = ident; /* ID */

rcvd_clear(ntransmitted+1);

if (timing) {
if (options & F_LATENCY) {
struct timeval tmp_tv;
gettimeofday(&tmp_tv, NULL);
memcpy(icp+1, &tmp_tv, sizeof(tmp_tv));
} else {
memset(icp+1, 0, sizeof(struct timeval));
}
}

cc = datalen + 8; /* skips ICMP portion */

/* compute ICMP checksum here */
icp->checksum = in_cksum((u_short *)icp, cc, 0);

if (timing && !(options & F_LATENCY)) {
struct timeval tmp_tv;
gettimeofday(&tmp_tv, NULL);
memcpy(icp+1, &tmp_tv, sizeof(tmp_tv));
icp->checksum = in_cksum((u_short *)&tmp_tv, sizeof(tmp_tv), ~icp->checksum);
}

do {
static struct iovec iov = {outpack, 0};
static struct msghdr m = {
.msg_name = &wheret,
.msg_namelen = sizeof(wheret),
.msg_iov = &iov,
.msg_iovlen = 1,
.msg_control = &cmsg,
.msg_controllen = 0,
.msg_flags = 0,
};
m.msg_controllen = cmsg_len;
iov.iov_len = cc;

i = sendmsg(icmp_sock, &m, confirm);
confirm = 0;
} while (0);

return (cc == i ? 0 : i);
}

```

```

/*
 * parse_reply --
 * Print out the packet, if it came from us. This logic is necessary
 * because ALL readers of the ICMP socket get a copy of ALL ICMP packets
 * which arrive ('tis only fair). This permits multiple copies of this
 * program to be run without having intermingled output (or statistics!).
 */
void pr_echo_reply(__u8 *_icp, int len)
{
    struct icmphdr *_icp = (struct icmphdr *)_icp;
    printf(" icmp_seq=%u", ntohs(icp->un.echo.sequence));
}

int
parse_reply(struct msghdr *msg, int cc, void *addr, struct timeval *tv)
{
    struct sockaddr_in *from = addr;
    __u8 *buf = msg->msg_iov->iov_base;
    struct icmphdr *icp;
    struct iphdr *ip;
    int hlen;
    int csfailed;
    struct cmsghdr *cmsg;
    int ttl;
    __u8 *opts;
    int optlen;

    /* Check the IP header */
    ip = (struct iphdr *)buf;
    if (!using_ping_socket) {
        hlen = ip->ihl*4;
        if (cc < hlen + 8 || ip->ihl < 5) {
            if (options & F_VERBOSE)
                fprintf(stderr, "ping: packet too short (%d bytes) from %s\n", cc,
                    pr_addr(from->sin_addr.s_addr));
            pr_addr(from->sin_addr.s_addr);
            return 1;
        }
        ttl = ip->ttl;
        opts = buf + sizeof(struct iphdr);
        optlen = hlen - sizeof(struct iphdr);
    } else {
        hlen = 0;
        ttl = 0;
        opts = buf;
        optlen = 0;
        for (cmsg = CMSG_FIRSTHDR(msg); cmsg; cmsg = CMSG_NXTHDR(msg, cmsg)) {
            if (cmsg->cmsg_level != SOL_IP)
                continue;
            if (cmsg->cmsg_type == IP_TTL) {
                if (cmsg->cmsg_len < sizeof(int))
                    continue;
                ttl = *(int *) CMSG_DATA(cmsg);
            } else if (cmsg->cmsg_type == IP_RETOPTS) {
                opts = (__u8 *) CMSG_DATA(cmsg);
            }
        }
    }
}

```

```

    optlen = cmsg->cmsg_len;
}
}
}

/* Now the ICMP part */
cc -= hlen;
icp = (struct icmphdr *) (buf + hlen);
csfailed = in_cksum((u_short *) icp, cc, 0);

if (icp->type == ICMP_ECHOREPLY) {
    if (!is_ours(icp->un.echo.id))
        return 1; /* 'Twas not our ECHO */
    if (gather_statistics((__u8 *) icp, sizeof(*icp), cc,
        ntohs(icp->un.echo.sequence),
        ttl, 0, tv, pr_addr(from->sin_addr.s_addr),
        pr_echo_reply))
        return 0;
} else {
    /* We fall here when a redirect or source quench arrived.
     * Also this branch processes icmp errors, when IP_RECVERR
     * is broken. */

    switch (icp->type) {
    case ICMP_ECHO:
        /* MUST NOT */
        return 1;
    case ICMP_SOURCE_QUENCH:
    case ICMP_REDIRECT:
    case ICMP_DEST_UNREACH:
    case ICMP_TIME_EXCEEDED:
    case ICMP_PARAMETERPROB:
        {
            struct iphdr * iph = (struct iphdr *) (&icp[1]);
            struct icmphdr * icp1 = (struct icmphdr *) ((unsigned char *) iph + iph->ihl*4);
            int error_pkt;
            if (cc < 8+sizeof(struct iphdr)+8 ||
                cc < 8+iph->ihl*4+8)
                return 1;
            if (icp1->type != ICMP_ECHO ||
                iph->daddr != where.to.sin_addr.s_addr ||
                !is_ours(icp1->un.echo.id))
                return 1;
            error_pkt = (icp->type != ICMP_REDIRECT &&
                icp->type != ICMP_SOURCE_QUENCH);
            if (error_pkt) {
                acknowledge(ntohs(icp1->un.echo.sequence));
                if (working_recverr) {
                    return 0;
                } else {
                    static int once;
                    /* Sigh, IP_RECVERR for raw socket
                     * was broken until 2.4.9. So, we ignore
                     * the first error and warn on the second.
                     */

```

```

if (once++ == 1)
fprintf(stderr, "\rWARNING: kernel is not very fresh, upgrade is recommended.\n");
if (once == 1)
return 0;
}
}
nerrors+=error_pkt;
if (options&F_QUIET)
return !error_pkt;
if (options & F_FLOOD) {
if (error_pkt)
write_stdout("\bE", 2);
return !error_pkt;
}
print_timestamp();
printf("From %s: icmp_seq=%u ",
      pr_addr(from->sin_addr.s_addr),
      ntohs(icp1->un.echo.sequence));
if (csfailed)
printf("(BAD CHECKSUM)");
pr_icmph(icp->type, icp->code, ntohl(icp->un.gateway), icp);
return !error_pkt;
}
default:
/* MUST NOT */
break;
}
if ((options & F_FLOOD) && !(options & (F_VERBOSE|F_QUIET))) {
if (!csfailed)
write_stdout("!E", 2);
else
write_stdout("!EC", 3);
return 0;
}
if (!(options & F_VERBOSE) || uid)
return 0;
if (options & F_PTIMEOFDAY) {
struct timeval recv_time;
gettimeofday(&recv_time, NULL);
printf("%lu.%06lu ", (unsigned long)recv_time.tv_sec, (unsigned long)recv_time.tv_usec);
}
printf("From %s: ", pr_addr(from->sin_addr.s_addr));
if (csfailed) {
printf("(BAD CHECKSUM)\n");
return 0;
}
pr_icmph(icp->type, icp->code, ntohl(icp->un.gateway), icp);
return 0;
}

if (options & F_AUDIBLE) {
putchar('\a');
if (options & F_FLOOD)
fflush(stdout);
}

```

```

    }
    if (!(options & F_FLOOD)) {
        pr_options(opts, optlen + sizeof(struct iphdr));
        putchar('\n');
        fflush(stdout);
    }
    return 0;
}

#if BYTE_ORDER == LITTLE_ENDIAN
# define ODDBYTE(v)(v)
#elif BYTE_ORDER == BIG_ENDIAN
# define ODDBYTE(v)((u_short)(v) << 8)
#else
# define ODDBYTE(v)htons((u_short)(v) << 8)
#endif

u_short
in_cksum(const u_short *addr, register int len, u_short csum)
{
    register int nleft = len;
    const u_short *w = addr;
    register u_short answer;
    register int sum = csum;

    /*
     * Our algorithm is simple, using a 32 bit accumulator (sum),
     * we add sequential 16 bit words to it, and at the end, fold
     * back all the carry bits from the top 16 bits into the lower
     * 16 bits.
     */
    while (nleft > 1) {
        sum += *w++;
        nleft -= 2;
    }

    /* mop up an odd byte, if necessary */
    if (nleft == 1)
        sum += ODDBYTE(*(u_char *)w); /* le16toh() may be unavailable on old systems */

    /*
     * add back carry outs from top 16 bits to low 16 bits
     */
    sum = (sum >> 16) + (sum & 0xffff); /* add hi 16 to low 16 */
    sum += (sum >> 16); /* add carry */
    answer = ~sum; /* truncate to 16 bits */
    return (answer);
}

/*
 * pr_icmph --
 * Print a descriptive string about an ICMP header.
 */
void pr_icmph(__u8 type, __u8 code, __u32 info, struct icmphdr *icp)

```



```
{
switch(type) {
case ICMP_ECHOREPLY:
printf("Echo Reply\n");
/* XXX ID + Seq + Data */
break;
case ICMP_DEST_UNREACH:
switch(code) {
case ICMP_NET_UNREACH:
printf("Destination Net Unreachable\n");
break;
case ICMP_HOST_UNREACH:
printf("Destination Host Unreachable\n");
break;
case ICMP_PROT_UNREACH:
printf("Destination Protocol Unreachable\n");
break;
case ICMP_PORT_UNREACH:
printf("Destination Port Unreachable\n");
break;
case ICMP_FRAG_NEEDED:
printf("Frag needed and DF set (mtu = %u)\n", info);
break;
case ICMP_SR_FAILED:
printf("Source Route Failed\n");
break;
case ICMP_NET_UNKNOWN:
printf("Destination Net Unknown\n");
break;
case ICMP_HOST_UNKNOWN:
printf("Destination Host Unknown\n");
break;
case ICMP_HOST_ISOLATED:
printf("Source Host Isolated\n");
break;
case ICMP_NET_ANO:
printf("Destination Net Prohibited\n");
break;
case ICMP_HOST_ANO:
printf("Destination Host Prohibited\n");
break;
case ICMP_NET_UNR_TOS:
printf("Destination Net Unreachable for Type of Service\n");
break;
case ICMP_HOST_UNR_TOS:
printf("Destination Host Unreachable for Type of Service\n");
break;
case ICMP_PKT_FILTERED:
printf("Packet filtered\n");
break;
case ICMP_PREC_VIOLATION:
printf("Precedence Violation\n");
break;
case ICMP_PREC_CUTOFF:
```

```

printf("Precedence Cutoff\n");
break;
default:
printf("Dest Unreachable, Bad Code: %d\n", code);
break;
}
if (icp && (options & F_VERBOSE))
pr_iph((struct iphdr*)(icp + 1));
break;
case ICMP_SOURCE_QUENCH:
printf("Source Quench\n");
if (icp && (options & F_VERBOSE))
pr_iph((struct iphdr*)(icp + 1));
break;
case ICMP_REDIRECT:
switch(code) {
case ICMP_REDIRECT_NET:
printf("Redirect Network");
break;
case ICMP_REDIRECT_HOST:
printf("Redirect Host");
break;
case ICMP_REDIRECT_NETTOS:
printf("Redirect Type of Service and Network");
break;
case ICMP_REDIRECT_HOSTTOS:
printf("Redirect Type of Service and Host");
break;
default:
printf("Redirect, Bad Code: %d", code);
break;
}
printf("(New nexthop: %s)\n",
pr_addr(icp ? icp->un.gateway : info));
if (icp && (options & F_VERBOSE))
pr_iph((struct iphdr*)(icp + 1));
break;
case ICMP_ECHO:
printf("Echo Request\n");
/* XXX ID + Seq + Data */
break;
case ICMP_TIME_EXCEEDED:
switch(code) {
case ICMP_EXC_TTL:
printf("Time to live exceeded\n");
break;
case ICMP_EXC_FRAGTIME:
printf("Frag reassembly time exceeded\n");
break;
default:
printf("Time exceeded, Bad Code: %d\n", code);
break;
}
if (icp && (options & F_VERBOSE))
pr_iph((struct iphdr*)(icp + 1));

```

```

break;
case ICMP_PARAMETERPROB:
printf("Parameter problem: pointer = %u\n", icp ? (ntohl(icp->un.gateway)>>24) : info);
if (icp && (options & F_VERBOSE))
pr_iph((struct iphdr*)(icp + 1));
break;
case ICMP_TIMESTAMP:
printf("Timestamp\n");
/* XXX ID + Seq + 3 timestamps */
break;
case ICMP_TIMESTAMPREPLY:
printf("Timestamp Reply\n");
/* XXX ID + Seq + 3 timestamps */
break;
case ICMP_INFO_REQUEST:
printf("Information Request\n");
/* XXX ID + Seq */
break;
case ICMP_INFO_REPLY:
printf("Information Reply\n");
/* XXX ID + Seq */
break;
#ifdef ICMP_MASKREQ
case ICMP_MASKREQ:
printf("Address Mask Request\n");
break;
#endif
#ifdef ICMP_MASKREPLY
case ICMP_MASKREPLY:
printf("Address Mask Reply\n");
break;
#endif
default:
printf("Bad ICMP type: %d\n", type);
}
}

void pr_options(unsigned char * cp, int hlen)
{
int i, j;
int optlen, totlen;
unsigned char * optptr;
static int old_rrlen;
static char old_rr[MAX_IPOPTLEN];

totlen = hlen - sizeof(struct iphdr);
optptr = cp;

while (totlen > 0) {
if (*optptr == IPOPT_EOL)
break;
if (*optptr == IPOPT_NOP) {
totlen--;
optptr++;
}
}

```

```

printf("\nNOP");
continue;
}
cp = optptr;
optlen = optptr[1];
if (optlen < 2 || optlen > totlen)
break;

switch (*cp) {
case IPOPT_SSRR:
case IPOPT_LSRR:
printf("\n%cSRR: ", *cp==IPOPT_SSRR ? 'S' : 'L');
j = *++cp;
i = *++cp;
i -= 4;
cp++;
if (j > IPOPT_MINOFF) {
for (;;) {
__u32 address;
memcpy(&address, cp, 4);
cp += 4;
if (address == 0)
printf("\t0.0.0.0");
else
printf("\t%s", pr_addr(address));
j -= 4;
putchar('\n');
if (j <= IPOPT_MINOFF)
break;
}
}
break;
case IPOPT_RR:
j = *++cp; /* get length */
i = *++cp; /* and pointer */
if (i > j)
i = j;
i -= IPOPT_MINOFF;
if (i <= 0)
break;
if (i == old_rlen
    && !memcmp(cp, old_rr, i)
    && !(options & F_FLOOD)) {
printf("\t(same route)");
i = ((i + 3) / 4) * 4;
cp += i;
break;
}
old_rlen = i;
memcpy(old_rr, (char *)cp, i);
printf("\nRR: ");
cp++;
for (;;) {
__u32 address;
memcpy(&address, cp, 4);

```

```

cp += 4;
if (address == 0)
printf("\t0.0.0.0");
else
printf("\t%s", pr_addr(address));
i -= 4;
putchar('\n');
if (i <= 0)
break;
}
break;
case IPOPT_TS:
{
int stdtime = 0, nonstdtime = 0;
__u8 flags;
j = *++cp; /* get length */
i = *++cp; /* and pointer */
if (i > j)
i = j;
i -= 5;
if (i <= 0)
break;
flags = *++cp;
printf("\nTS: ");
cp++;
for (;;) {
long l;

if ((flags&0xF) != IPOPT_TS_TSONLY) {
__u32 address;
memcpy(&address, cp, 4);
cp += 4;
if (address == 0)
printf("\t0.0.0.0");
else
printf("\t%s", pr_addr(address));
i -= 4;
if (i <= 0)
break;
}
l = *cp++;
l = (l<<8) + *cp++;
l = (l<<8) + *cp++;
l = (l<<8) + *cp++;

if (l & 0x80000000) {
if (nonstdtime==0)
printf("\t%ld absolute not-standard", l&0x7fffffff);
else
printf("\t%ld not-standard", (l&0x7fffffff) - nonstdtime);
nonstdtime = l&0x7fffffff;
} else {
if (stdtime==0)
printf("\t%ld absolute", l);

```

```

else
printf("\t%d", l - stdtime);
stdtime = l;
}
i -= 4;
putchar('\n');
if (i <= 0)
break;
}
if (flags>>4)
printf("Unrecorded hops: %d\n", flags>>4);
break;
}
default:
printf("\nunknown option %x", *cp);
break;
}
totlen -= optlen;
optptr += optlen;
}
}

/*
 * pr_iph --
 * Print an IP header with options.
 */
void pr_iph(struct iphdr *ip)
{
int hlen;
u_char *cp;

hlen = ip->ihl << 2;
cp = (u_char *)ip + 20; /* point to options */

printf("Vr HL TOS Len  ID Flg  off TTL Pro  cks    Src    Dst Data\n");
printf(" %1x %1x %02x %04x %04x",
    ip->version, ip->ihl, ip->tos, ip->tot_len, ip->id);
printf("  %1x %04x", ((ip->frag_off) & 0xe000) >> 13,
    (ip->frag_off) & 0x1fff);
printf(" %02x %02x %04x", ip->ttl, ip->protocol, ip->check);
printf(" %s ", inet_ntoa(*(struct in_addr *)&ip->saddr));
printf(" %s ", inet_ntoa(*(struct in_addr *)&ip->daddr));
printf("\n");
pr_options(cp, hlen);
}

/*
 * pr_addr --
 * Return an ascii host address as a dotted quad and optionally with
 * a hostname.
 */
char *
pr_addr(__u32 addr)
{

```

```

struct hostent *hp;
static char buf[4096];

in_pr_addr = !setjmp(pr_addr_jmp);

if (exiting || (options & F_NUMERIC) ||
    !(hp = gethostbyaddr((char *)&addr, 4, AF_INET)))
    sprintf(buf, "%s", inet_ntoa(*(struct in_addr *)&addr));
else {
    char *s;
    #if USE_IDN
    if (idna_to_unicode_lzlz(hp->h_name, &s, 0) != IDNA_SUCCESS)
        s = NULL;
    #else
    s = NULL;
    #endif
    snprintf(buf, sizeof(buf), "%s (%s)", s ? s : hp->h_name,
        inet_ntoa(*(struct in_addr *)&addr));
    #if USE_IDN
    free(s);
    #endif
}

in_pr_addr = 0;

return(buf);
}

/* Set Type of Service (TOS) and other Quality of Service relating bits */
int parsetos(char *str)
{
    const char *cp;
    int tos;
    char *ep;

    /* handle both hex and decimal values */
    if (str[0] == '0' && (str[1] == 'x' || str[1] == 'X')) {
        cp = str + 2;
        tos = (int)strtol(cp, &ep, 16);
    } else
        tos = (int)strtol(str, &ep, 10);

    /* doesn't look like decimal or hex, eh? */
    if (*ep != '\0') {
        fprintf(stderr, "ping: \"%s\" bad value for TOS\n", str);
        exit(2);
    }

    if (tos > TOS_MAX) {
        fprintf(stderr, "ping: the decimal value of TOS bits must be 0-254 (or zero)\n");
        exit(2);
    }
    return(tos);
}

```

```

}

#include <linux/filter.h>

void install_filter(void)
{
    static int once;
    static struct sock_filter insns[] = {
        BPF_STMT(BPF_LDX|BPF_B|BPF_MSH, 0), /* Skip IP header. F..g BSD... Look into ping6. */
        BPF_STMT(BPF_LD|BPF_H|BPF_IND, 4), /* Load icmp echo ident */
        BPF_JUMP(BPF_JMP|BPF_JEQ|BPF_K, 0xAAAA, 0, 1), /* Ours? */
        BPF_STMT(BPF_RET|BPF_K, ~0U), /* Yes, it passes. */
        BPF_STMT(BPF_LD|BPF_B|BPF_IND, 0), /* Load icmp type */
        BPF_JUMP(BPF_JMP|BPF_JEQ|BPF_K, ICMP_ECHOREPLY, 1, 0), /* Echo? */
        BPF_STMT(BPF_RET|BPF_K, 0xFFFFFFFF), /* No. It passes. */
        BPF_STMT(BPF_RET|BPF_K, 0) /* Echo with wrong ident. Reject. */
    };
    static struct sock_fprog filter = {
        sizeof insns / sizeof(insns[0]),
        insns
    };

    if (once)
        return;
    once = 1;

    /* Patch bpflet for current identifier. */
    insns[2] = (struct sock_filter)BPF_JUMP(BPF_JMP|BPF_JEQ|BPF_K, htons(ident), 0, 1);

    if (setsockopt(icmp_sock, SOL_SOCKET, SO_ATTACH_FILTER, &filter, sizeof(filter)))
        perror("WARNING: failed to install socket filter\n");
}

#define USAGE_NEWLINE"\n      "

void usage(void)
{
    fprintf(stderr,
        "Usage: ping"
        "\n[-"
        "aAbBdDfhLnOqrRUvV"
        "]"
        "\n[-c count]"
        "\n[-i interval]"
        "\n[-I interface]"
        USAGE_NEWLINE
        "\n[-m mark]"
        "\n[-M pmtudisc_option]"
        "\n[-l preload]"
        "\n[-p pattern]"
        "\n[-Q tos]"
        USAGE_NEWLINE
        "\n[-s packetsize]"
        "\n[-S sndbuf]"
        "\n[-t ttl]"
    );
}

```



```

" [-T timestamp_option]"
USAGE_NEWLINE
" [-w deadline]"
" [-W timeout]"
" [hop1 ...] destination"
"\n"
);
exit(2);
}
/*
 * Copyright (c) 1983 Regents of the University of California.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 * This product includes software developed by the University of
 * California, Berkeley and its contributors.
 * 4. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#ifdef lint
char copyright[] =
"@(#) Copyright (c) 1983 Regents of the University of California.\n\
All rights reserved.\n";
#endif /* not lint */

#ifdef lint
/*static char scsid[] = "from: @(#)tftpd.c5.13 (Berkeley) 2/26/91";*/
/*static char rsid[] = "$Id: tftpd.c,v 1.3 1993/08/01 18:28:53 mycroft Exp $";*/
#endif /* not lint */

/*

```

```

* Trivial file transfer protocol server.
*
* This version includes many modifications by Jim Guyton <guyton@rand-unix>
*/

#include <sys/types.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <unistd.h>
#include <signal.h>
#include <fcntl.h>

#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#include <setjmp.h>
#include <syslog.h>
#include <stdio.h>
#include <errno.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>

#include "tftp.h"

#ifndef MSG_CONFIRM
#define MSG_CONFIRM 0
#warning Please, upgrade kernel, otherwise this tftpd has no advantages.
#endif

#define TIMEOUT5

intpeer;
intrexmtval = TIMEOUT;
intmaxtimeout = 5*TIMEOUT;

#definePKTSIZESEGSIZE+4
charbuf[PKTSIZE];
charackbuf[PKTSIZE];
union {
    structsockaddr sa;
    structsockaddr_in sin;
    structsockaddr_in6 sin6;
} from;
socklen_tfromlen;

#define MAXARG1
char*dirs[MAXARG+1];

void tftp(struct tftphdr *tp, int size) __attribute__((noreturn));
void nak(int error);
int validate_access(char *filename, int mode);

struct formats;

```

```

void sendfile(struct formats *pf);
void recvfile(struct formats *pf);

int main(int ac, char **av)
{
    register struct tftphdr *tp;
    register int n = 0;
    int on = 1;

    /* Sanity. If parent forgot to setuid() on us. */
    if (geteuid() == 0) {
        if (setgid(65534)) {
            syslog(LOG_ERR, "setgid");
            exit(1);
        }
        if (setuid(65534)) {
            syslog(LOG_ERR, "setuid");
            exit(1);
        }
    }

    ac--; av++;
    while (ac-- > 0 && n < MAXARG)
        dirs[n++] = *av++;

    openlog("tftpd", LOG_PID, LOG_DAEMON);
    if (ioctl(0, FIONBIO, &on) < 0) {
        syslog(LOG_ERR, "ioctl(FIONBIO): %m\n");
        exit(1);
    }
    fromlen = sizeof (from);
    n = recvfrom(0, buf, sizeof (buf), 0,
        (struct sockaddr *)&from, &fromlen);
    if (n < 0) {
        if (errno != EAGAIN)
            syslog(LOG_ERR, "recvfrom: %m\n");
        exit(1);
    }
    /*
     * Now that we have read the message out of the UDP
     * socket, we fork and exit. Thus, inetd will go back
     * to listening to the tftp port, and the next request
     * to come in will start up a new instance of tftpd.
     *
     * We do this so that inetd can run tftpd in "wait" mode.
     * The problem with tftpd running in "nowait" mode is that
     * inetd may get one or more successful "selects" on the
     * tftp port before we do our receive, so more than one
     * instance of tftpd may be started up. Worse, if tftpd
     * break before doing the above "recvfrom", inetd would
     * spawn endless instances, clogging the system.
     */

```

```

{
int pid;
int i;
socklen_t j;

for (i = 1; i < 20; i++) {
    pid = fork();
    if (pid < 0) {
sleep(i);
/*
 * flush out to most recently sent request.
 *
 * This may drop some request, but those
 * will be resent by the clients when
 * they timeout. The positive effect of
 * this flush is to (try to) prevent more
 * than one tftpd being started up to service
 * a single request from a single client.
 */
j = sizeof from;
i = recvfrom(0, buf, sizeof (buf), 0,
    (struct sockaddr *)&from, &j);
if (i > 0) {
n = i;
fromlen = j;
}
    } else {
break;
    }
}
if (pid < 0) {
syslog(LOG_ERR, "fork: %m\n");
exit(1);
} else if (pid != 0) {
exit(0);
}
}
alarm(0);
close(0);
close(1);
peer = socket(from.sa.sa_family, SOCK_DGRAM, 0);
if (peer < 0) {
syslog(LOG_ERR, "socket: %m\n");
exit(1);
}
if (connect(peer, (struct sockaddr *)&from, sizeof(from)) < 0) {
syslog(LOG_ERR, "connect: %m\n");
exit(1);
}
tp = (struct tftphdr *)buf;
tp->th_opcode = ntohs(tp->th_opcode);
if (tp->th_opcode == RRQ || tp->th_opcode == WRQ)
tftp(tp, n);
exit(1);
}

```

```

struct formats {
char*f_mode;
int(*f_validate)(char *filename, int mode);
void(*f_send)(struct formats*);
void(*f_rcv)(struct formats*);
intf_convert;
} formats[] = {
{ "netascii",validate_access,sendfile,recvfile, 1 },
{ "octet",validate_access,sendfile,recvfile, 0 },
#ifdef notdef
{ "mail",validate_user,sendmail,recvmail, 1 },
#endif
{ 0 }
};

/*
 * Handle initial connection protocol.
 */
void tftp(struct tftphdr *tp, int size)
{
register char *cp;
int first = 1, ecode;
register struct formats *pf;
char *filename, *mode = NULL;

filename = cp = tp->th_stuff;
again:
while (cp < buf + size) {
if (*cp == '\0')
break;
cp++;
}
if (*cp != '\0') {
nak(EBADOP);
exit(1);
}
if (first) {
mode = ++cp;
first = 0;
goto again;
}
for (cp = mode; *cp; cp++)
if (isupper(*cp))
*cp = tolower(*cp);
for (pf = formats; pf->f_mode; pf++)
if (strcmp(pf->f_mode, mode) == 0)
break;
if (pf->f_mode == 0) {
nak(EBADOP);
exit(1);
}
ecode = (*pf->f_validate)(filename, tp->th_opcode);
if (ecode) {

```

```

    nak(ecode);
    exit(1);
}
if (tp->th_opcode == WRQ)
    (*pf->f_rcv)(pf);
else
    (*pf->f_send)(pf);
    exit(0);
}

FILE *file;

/*
 * Validate file access. Since we
 * have no uid or gid, for now require
 * file to exist and be publicly
 * readable/writable.
 * If we were invoked with arguments
 * from inetd then the file must also be
 * in one of the given directory prefixes.
 * Note also, full path name must be
 * given as we have no login directory.
 */
int validate_access(char *filename, int mode)
{
    struct stat stbuf;
    int fd;
    char *cp;
    char fnamebuf[1024+512];

    for (cp = filename; *cp; cp++) {
        if(*cp == '.' && (cp == filename || strcmp(cp-1, "../", 4) == 0)) {
            syslog(LOG_ERR, "bad path %s", filename);
            return(EACCESS);
        }
    }

    if (*filename == '/')
        filename++;

    if (!*dirs) {
        syslog(LOG_ERR, "no dirs");
        return EACCESS;
    }
    snprintf(fnamebuf, sizeof(fnamebuf)-1, "%s/%s", *dirs, filename);
    filename = fnamebuf;

    if (stat(filename, &stbuf) < 0) {
        syslog(LOG_ERR, "stat %s : %m", filename);
        return (errno == ENOENT ? ENOTFOUND : EACCESS);
    }
    if (mode == RRQ) {
        if ((stbuf.st_mode & (S_IREAD >> 6)) == 0) {
            syslog(LOG_ERR, "not readable %s", filename);

```

```

return (EACCESS);
}
} else {
if ((stbuf.st_mode & (S_IWRITE >> 6)) == 0) {
syslog(LOG_ERR, "not writable %s", filename);
return (EACCESS);
}
}
fd = open(filename, mode == RRQ ? 0 : 1);
if (fd < 0) {
syslog(LOG_ERR, "cannot open %s: %m", filename);
return (errno + 100);
}
file = fdopen(fd, (mode == RRQ) ? "r" : "w");
if (file == NULL) {
return errno + 100;
}
return (0);
}

intconfirmed;
inttimeout;
jmp_buf timeoutbuf;

void timer(int signo)
{
confirmed = 0;
timeout += rexmtval;
if (timeout >= maxtimeout)
exit(1);
longjmp(timeoutbuf, 1);
}

/*
 * Send the requested file.
 */
void sendfile(struct formats *pf)
{
struct tftphdr *dp;
register struct tftphdr *ap; /* ack packet */
volatile int block = 1;
int size, n;

confirmed = 0;
signal(SIGALRM, timer);
dp = r_init();
ap = (struct tftphdr *)ackbuf;
do {
size = readit(file, &dp, pf->f_convert);
if (size < 0) {
nak(errno + 100);
goto abort;
}
dp->th_opcode = htons((u_short)DATA);

```

```

dp->th_block = htons((u_short)block);
timeout = 0;
(void) setjmp(timeoutbuf);

send_data:
if (send(peer, dp, size + 4, confirmed) != size + 4) {
    syslog(LOG_ERR, "tftpd: write: %m\n");
    goto abort;
}
confirmed = 0;
read_ahead(file, pf->f_convert);
for ( ; ; ) {
    alarm(rexmtval);    /* read the ack */
    n = recv(peer, ackbuf, sizeof(ackbuf), 0);
    alarm(0);
    if (n < 0) {
        syslog(LOG_ERR, "tftpd: read: %m\n");
        goto abort;
    }
    ap->th_opcode = ntohs((u_short)ap->th_opcode);
    ap->th_block = ntohs((u_short)ap->th_block);

    if (ap->th_opcode == ERROR)
        goto abort;

    if (ap->th_opcode == ACK) {
        if (ap->th_block == block) {
            confirmed = MSG_CONFIRM;
            break;
        }
        /* Re-synchronize with the other side */
        synchnet(peer);
        if (ap->th_block == (block - 1)) {
            goto send_data;
        }
    }

    block++;
} while (size == SEGSIZE);
abort:
(void) fclose(file);
}

void justquit(int signo)
{
    exit(0);
}

/*
 * Receive a file.
 */
void recvfile(struct formats *pf)
{

```



```

struct tftphdr *dp;
register struct tftphdr *ap; /* ack buffer */
volatile int block = 0, n, size;

confirmed = 0;
signal(SIGALRM, timer);
dp = w_init();
ap = (struct tftphdr *)ackbuf;
do {
    timeout = 0;
    ap->th_opcode = htons((u_short)ACK);
    ap->th_block = htons((u_short)block);
    block++;
    (void) setjmp(timeoutbuf);
    send_ack:
    if (send(peer, ackbuf, 4, confirmed) != 4) {
        syslog(LOG_ERR, "tftpd: write: %m\n");
        goto abort;
    }
    confirmed = 0;
    write_behind(file, pf->f_convert);
    for ( ; ) {
        alarm(rexmtval);
        n = recv(peer, dp, PKTSIZE, 0);
        alarm(0);
        if (n < 0) { /* really? */
            syslog(LOG_ERR, "tftpd: read: %m\n");
            goto abort;
        }
        dp->th_opcode = ntohs((u_short)dp->th_opcode);
        dp->th_block = ntohs((u_short)dp->th_block);
        if (dp->th_opcode == ERROR)
            goto abort;
        if (dp->th_opcode == DATA) {
            if (dp->th_block == block) {
                confirmed = MSG_CONFIRM;
                break; /* normal */
            }
            /* Re-synchronize with the other side */
            (void) synchnet(peer);
            if (dp->th_block == (block-1))
                goto send_ack; /* rexmit */
        }
        /* size = write(file, dp->th_data, n - 4); */
        size = writeit(file, &dp, n - 4, pf->f_convert);
        if (size != (n-4)) { /* ahem */
            if (size < 0) nak(errno + 100);
            else nak(ENOSPACE);
            goto abort;
        }
    } while (size == SEGSIZE);
    write_behind(file, pf->f_convert);
    (void) fclose(file); /* close data file */
}

```

```

ap->th_opcode = htons((u_short)ACK); /* send the "final" ack */
ap->th_block = htons((u_short)(block));
(void) send(peer, ackbuf, 4, confirmed);

signal(SIGALRM, justquit); /* just quit on timeout */
alarm(rexmtval);
n = recv(peer, buf, sizeof (buf), 0); /* normally times out and quits */
alarm(0);
if (n >= 4 && /* if read some data */
    dp->th_opcode == DATA && /* and got a data block */
    block == dp->th_block) { /* then my last ack was lost */
(void) send(peer, ackbuf, 4, 0); /* resend final ack */
}
abort:
return;
}

struct errmsg {
int e_code;
char *e_msg;
} errmsgs[] = {
{ EUNDEF, "Undefined error code" },
{ ENOTFOUND, "File not found" },
{ EACCESS, "Access violation" },
{ ENOSPACE, "Disk full or allocation exceeded" },
{ EBADOP, "Illegal TFTP operation" },
{ EBADID, "Unknown transfer ID" },
{ EEXISTS, "File already exists" },
{ ENOUSER, "No such user" },
{ -1, 0 }
};

/*
 * Send a nak packet (error message).
 * Error code passed in is one of the
 * standard TFTP codes, or a UNIX errno
 * offset by 100.
 */
void nak(int error)
{
register struct tftphdr *tp;
int length;
register struct errmsg *pe;

tp = (struct tftphdr *)buf;
tp->th_opcode = htons((u_short)ERROR);
tp->th_code = htons((u_short)error);
for (pe = errmsgs; pe->e_code >= 0; pe++)
if (pe->e_code == error)
break;
if (pe->e_code < 0) {
pe->e_msg = strerror(error - 100);
tp->th_code = EUNDEF; /* set 'undef' errorcode */
}
}

```

```

strcpy(tp->th_msg, pe->e_msg);
length = strlen(pe->e_msg);
tp->th_msg[length] = '\0';
length += 5;
if (send(peer, buf, length, 0) != length)
syslog(LOG_ERR, "nak: %m\n");
}
/*
 * tracepath.c
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version
 * 2 of the License, or (at your option) any later version.
 *
 * Authors: Alexey Kuznetsov, <kuznet@ms2.inr.ac.ru>
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <limits.h>
#include <time.h>
#include <sys/socket.h>
#include <linux/types.h>
#include <linux/errqueue.h>
#include <errno.h>
#include <string.h>
#include <netdb.h>
#include <netinet/in.h>
#include <resolv.h>
#include <sys/time.h>
#include <sys/uio.h>
#include <arpa/inet.h>
#ifdef USE_IDN
#include <idna.h>
#include <locale.h>
#endif

#ifdef IP_PMTUDISC_PROBE
#define IP_PMTUDISC_PROBE3
#endif

#define MAX_HOPS_LIMIT255
#define MAX_HOPS_DEFAULT30

struct hhistory
{
    inthops;
    struct timeval sendtime;
};

struct hhistory his[64];
int hisptr;

```

```

struct sockaddr_in target;
__u16 base_port;
int max_hops = MAX_HOPS_DEFAULT;

const int overhead = 28;
int mtu = 65535;
void *pktbuf;
int hops_to = -1;
int hops_from = -1;
int no_resolve = 0;
int show_both = 0;

#define HOST_COLUMN_SIZE 52

struct probehdr
{
    __u32 ttl;
    struct timeval tv;
};

void data_wait(int fd)
{
    fd_set fds;
    struct timeval tv;
    FD_ZERO(&fds);
    FD_SET(fd, &fds);
    tv.tv_sec = 1;
    tv.tv_usec = 0;
    select(fd+1, &fds, NULL, NULL, &tv);
}

void print_host(const char *a, const char *b, int both)
{
    int plen;
    plen = printf("%s", a);
    if (both)
        plen += printf(" (%s)", b);
    if (plen >= HOST_COLUMN_SIZE)
        plen = HOST_COLUMN_SIZE - 1;
    printf("%*s", HOST_COLUMN_SIZE - plen, "");
}

int recverr(int fd, int ttl)
{
    int res;
    struct probehdr rcvbuf;
    char cbuf[512];
    struct iovec iov;
    struct msghdr msg;
    struct cmsghdr *cmsg;
    struct sock_extended_err *e;
    struct sockaddr_in addr;
    struct timeval tv;
    struct timeval *rettv;

```

```

int slot;
int rethops;
int sndhops;
int progress = -1;
int broken_router;

restart:
memset(&rcvbuf, -1, sizeof(rcvbuf));
iov.iov_base = &rcvbuf;
iov.iov_len = sizeof(rcvbuf);
msg.msg_name = (__u8*)&addr;
msg.msg_namelen = sizeof(addr);
msg.msg_iov = &iov;
msg.msg_iovlen = 1;
msg.msg_flags = 0;
msg.msg_control = cbuf;
msg.msg_controllen = sizeof(cbuf);

gettimeofday(&tv, NULL);
res = recvmsg(fd, &msg, MSG_ERRQUEUE);
if (res < 0) {
if (errno == EAGAIN)
return progress;
goto restart;
}

progress = mtu;

rethops = -1;
sndhops = -1;
e = NULL;
rettv = NULL;
slot = ntohs(addr.sin_port) - base_port;
if (slot >= 0 && slot < 63 && his[slot].hops) {
sndhops = his[slot].hops;
rettv = &his[slot].sendtime;
his[slot].hops = 0;
}
broken_router = 0;
if (res == sizeof(rcvbuf)) {
if (rcvbuf.ttl == 0 || rcvbuf.tv.tv_sec == 0) {
broken_router = 1;
} else {
sndhops = rcvbuf.ttl;
rettv = &rcvbuf.tv;
}
}

for (cmsg = CMSG_FIRSTHDR(&msg); cmsg; cmsg = CMSG_NXTHDR(&msg, cmsg)) {
if (cmsg->cmsg_level == SOL_IP) {
if (cmsg->cmsg_type == IP_RECVERR) {
e = (struct sock_extended_err *) CMSG_DATA(cmsg);
} else if (cmsg->cmsg_type == IP_TTL) {
memcpy(&rethops, CMSG_DATA(cmsg), sizeof(rethops));

```

```

    } else {
    printf("cmsg:%d\n ", cmsg->cmsg_type);
    }
    }
    }
    if (e == NULL) {
    printf("no info\n");
    return 0;
    }
    if (e->ee_origin == SO_EE_ORIGIN_LOCAL) {
    printf("%2d?: %*s ", ttl, -(HOST_COLUMN_SIZE - 1), "[LOCALHOST]");
    } else if (e->ee_origin == SO_EE_ORIGIN_ICMP) {
    char abuf[128];
    struct sockaddr_in *sin = (struct sockaddr_in*)(e+1);
    struct hostent *h = NULL;
    char *idn = NULL;

    inet_ntop(AF_INET, &sin->sin_addr, abuf, sizeof(abuf));

    if (sndhops>0)
    printf("%2d: ", sndhops);
    else
    printf("%2d?: ", ttl);

    if (!no_resolve || show_both) {
    fflush(stdout);
    h = gethostbyaddr((char *) &sin->sin_addr, sizeof(sin->sin_addr), AF_INET);
    }

    #ifndef USE_IDN
    if (h && idna_to_unicode_lzlz(h->h_name, &idn, 0) != IDNA_SUCCESS)
    idn = NULL;
    #endif
    if (no_resolve)
    print_host(abuf, h ? (idn ? idn : h->h_name) : abuf, show_both);
    else
    print_host(h ? (idn ? idn : h->h_name) : abuf, abuf, show_both);

    #ifndef USE_IDN
    free(idn);
    #endif
    }

    if (rettv) {
    int diff = (tv.tv_sec-rettv->tv_sec)*1000000+(tv.tv_usec-rettv->tv_usec);
    printf("%3d.%03dms ", diff/1000, diff%1000);
    if (broken_router)
    printf("(This broken router returned corrupted payload) ");
    }

    if (rethops >= 0) {
    if (rethops <= 64)
    rethops = 65-rethops;
    else if (rethops <= 128)
    rethops = 129-rethops;

```

```
else
rethops = 256-rethops;
}

switch (e->ee_erno) {
case ETIMEDOUT:
printf("\n");
break;
case EMSGSIZE:
printf("pmtu %d\n", e->ee_info);
mtu = e->ee_info;
progress = mtu;
break;
case ECONNREFUSED:
printf("reached\n");
hops_to = sndhops<0 ? ttl : sndhops;
hops_from = rethops;
return 0;
case EPROTO:
printf("!P\n");
return 0;
case EHOSTUNREACH:
if (e->ee_origin == SO_EE_ORIGIN_ICMP &&
    e->ee_type == 11 &&
    e->ee_code == 0) {
if (rethops>=0) {
if (sndhops>=0 && rethops != sndhops)
printf("asymm %2d ", rethops);
else if (sndhops<0 && rethops != ttl)
printf("asymm %2d ", rethops);
}
printf("\n");
break;
}
printf("!H\n");
return 0;
case ENETUNREACH:
printf("!N\n");
return 0;
case EACCES:
printf("!A\n");
return 0;
default:
printf("\n");
errno = e->ee_erno;
perror("NET ERROR");
return 0;
}
goto restart;
}

int probe_ttl(int fd, int ttl)
{
int i;
```

```

struct probehdr *hdr = pktbuf;

memset(pktbuf, 0, mtu);
restart:
for (i=0; i<10; i++) {
int res;

hdr->ttl = ttl;
target.sin_port = htons(base_port + hisptr);
gettimeofday(&hdr->tv, NULL);
his[hisptr].hops = ttl;
his[hisptr].sendtime = hdr->tv;
if (sendto(fd, pktbuf, mtu-overhead, 0, (struct sockaddr*)&target, sizeof(target)) > 0)
break;
res = recvrr(fd, ttl);
his[hisptr].hops = 0;
if (res==0)
return 0;
if (res > 0)
goto restart;
}
hisptr = (hisptr + 1)&63;

if (i<10) {
data_wait(fd);
if (recv(fd, pktbuf, mtu, MSG_DONTWAIT) > 0) {
printf("%2d?: reply received 8)\n", ttl);
return 0;
}
return recvrr(fd, ttl);
}

printf("%2d: send failed\n", ttl);
return 0;
}

static void usage(void) __attribute__((noreturn));

static void usage(void)
{
fprintf(stderr, "Usage: tracepath [-n] [-b] [-l <len>] [-p port] <destination>\n");
exit(-1);
}

int
main(int argc, char **argv)
{
struct hostent *he;
int fd;
int on;
int ttl;
char *p;
int ch;
#ifdef USE_IDN
int rc;

```



```
setlocale(LC_ALL, "");
#endif

while ((ch = getopt(argc, argv, "nbh?l:m:p:")) != EOF) {
switch(ch) {
case 'n':
no_resolve = 1;
break;
case 'b':
show_both = 1;
break;
case 'l':
if ((mtu = atoi(optarg)) <= overhead) {
fprintf(stderr, "Error: pktlen must be > %d and <= %d.\n",
overhead, INT_MAX);
exit(1);
}
break;
case 'm':
max_hops = atoi(optarg);
if (max_hops < 0 || max_hops > MAX_HOPS_LIMIT) {
fprintf(stderr,
"Error: max hops must be 0 .. %d (inclusive).\n",
MAX_HOPS_LIMIT);
}
break;
case 'p':
base_port = atoi(optarg);
break;
default:
usage();
}
}

argc -= optind;
argv += optind;

if (argc != 1)
usage();

fd = socket(AF_INET, SOCK_DGRAM, 0);
if (fd < 0) {
perror("socket");
exit(1);
}
target.sin_family = AF_INET;

/* Backward compatibility */
if (!base_port) {
p = strchr(argv[0], '/');
if (p) {
*p = 0;
base_port = atoi(p+1);
} else
```

```

base_port = 44444;
}

p = argv[0];
#ifdef USE_IDN
rc = idna_to_ascii_lz(argv[0], &p, 0);
if (rc != IDNA_SUCCESS) {
fprintf(stderr, "IDNA encoding failed: %s\n", idna_strerror(rc));
exit(2);
}
#endif

he = gethostbyname2(argv[0], AF_INET);
if (he == NULL) {
herror("gethostbyname2");
exit(1);
}

#ifdef USE_IDN
free(p);
#endif

memcpy(&target.sin_addr, he->h_addr, 4);

on = IP_PMTUDISC_PROBE;
if (setsockopt(fd, SOL_IP, IP_MTU_DISCOVER, &on, sizeof(on)) &&
    (on = IP_PMTUDISC_DO,
     setsockopt(fd, SOL_IP, IP_MTU_DISCOVER, &on, sizeof(on)))) {
perror("IP_MTU_DISCOVER");
exit(1);
}
on = 1;
if (setsockopt(fd, SOL_IP, IP_RECVERR, &on, sizeof(on))) {
perror("IP_RECVERR");
exit(1);
}
if (setsockopt(fd, SOL_IP, IP_RECVTTL, &on, sizeof(on))) {
perror("IP_RECVTTL");
exit(1);
}

pktbuf = malloc(mtu);
if (!pktbuf) {
perror("malloc");
exit(1);
}

for (ttl = 1; ttl <= max_hops; ttl++) {
int res;
int i;

on = ttl;
if (setsockopt(fd, SOL_IP, IP_TTL, &on, sizeof(on))) {
perror("IP_TTL");
exit(1);
}

```

```
}

restart:
for (i=0; i<3; i++) {
int old_mtu;

old_mtu = mtu;
res = probe_ttl(fd, ttl);
if (mtu != old_mtu)
goto restart;
if (res == 0)
goto done;
if (res > 0)
break;
}

if (res < 0)
printf("%2d: no reply\n", ttl);
}
printf("    Too many hops: pmtu %d\n", mtu);
done:
printf("    Resume: pmtu %d ", mtu);
if (hops_to>=0)
printf("hops %d ", hops_to);
if (hops_from>=0)
printf("back %d ", hops_from);
printf("\n");
exit(0);
}
```

1.56 keyutils

1.56.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder say-

ing it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geograph-

ical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w`. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c`
for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.57 kmod

1.57.1 Available under license

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public

License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception,

the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent

infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free

Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.58 krb5

1.58.1 Available under license

Copyright (C) 1985-2019 by the Massachusetts Institute of Technology.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Downloading of this software may constitute an export of cryptographic software from the United States of America that is subject to the United States Export Administration Regulations (EAR), 15 CFR 730-774. Additional laws or regulations may apply. It is the responsibility of the person or entity contemplating export to comply with all applicable export laws and regulations, including obtaining any required license from the U.S. government.

The U.S. government prohibits export of encryption source code to certain countries and individuals, including, but not limited to, the countries of Cuba, Iran, North Korea, Sudan, Syria, and residents and nationals of those countries.

Documentation components of this software distribution are licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License. (<https://creativecommons.org/licenses/by-sa/3.0/>)

Individual source code files are copyright MIT, Cygnus Support, Novell, OpenVision Technologies, Oracle, Red Hat, Sun Microsystems, FundsXpress, and others.

Project Athena, Athena, Athena MUSE, Discuss, Hesiod, Kerberos, Moira, and Zephyr are trademarks of the Massachusetts Institute of Technology (MIT). No commercial use of these trademarks may be made without prior written permission of MIT.

"Commercial use" means use of a name in a product or other for-profit manner. It does NOT prevent a commercial firm from referring to the MIT trademarks in order to convey information (although in doing so, recognition of their trademark status should be given).

=====

The following copyright and permission notice applies to the OpenVision Kerberos Administration system located in "kadmin/create", "kadmin/dbutil", "kadmin/passwd", "kadmin/server", "lib/kadm5", and portions of "lib/rpc":

Copyright, OpenVision Technologies, Inc., 1993-1996, All Rights Reserved

WARNING: Retrieving the OpenVision Kerberos Administration system source code, as described below, indicates your acceptance of the following terms. If you do not agree to the following terms, do not retrieve the OpenVision Kerberos administration system.

You may freely use and distribute the Source Code and Object Code compiled from it, with or without modification, but this Source Code is provided to you "AS IS" EXCLUSIVE OF ANY WARRANTY, INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY OTHER WARRANTY, WHETHER EXPRESS OR IMPLIED. IN NO EVENT WILL OPENVISION HAVE ANY LIABILITY FOR ANY LOST PROFITS, LOSS OF DATA OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, INCLUDING, WITHOUT LIMITATION, THOSE RESULTING FROM THE USE OF THE SOURCE CODE, OR THE FAILURE OF THE SOURCE CODE TO PERFORM, OR FOR ANY OTHER REASON.

OpenVision retains all copyrights in the donated Source Code. OpenVision also retains copyright to derivative works of the Source Code, whether created by OpenVision or by a third party. The OpenVision copyright notice must be preserved if derivative works are made based on the donated Source Code.

OpenVision Technologies, Inc. has donated this Kerberos Administration system to MIT for inclusion in the standard Kerberos 5 distribution. This donation underscores our commitment to continuing Kerberos technology development and our gratitude for the valuable work which has been performed by MIT and the Kerberos community.

=====

Portions contributed by Matt Crawford "crawd@fnal.gov" were work

performed at Fermi National Accelerator Laboratory, which is operated by Universities Research Association, Inc., under contract DE-AC02-76CHO3000 with the U.S. Department of Energy.

=====

Portions of "src/lib/crypto" have the following copyright:

Copyright (C) 1998 by the FundsXpress, INC.

All rights reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of FundsXpress. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. FundsXpress makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

=====

The implementation of the AES encryption algorithm in "src/lib/crypto/builtin/aes" has the following copyright:

Copyright (C) 2001, Dr Brian Gladman "brg@gladman.uk.net", Worcester, UK.
All rights reserved.

LICENSE TERMS

The free distribution and use of this software in both source and binary form is allowed (with or without changes) provided that:

1. distributions of this source code include the above copyright notice, this list of conditions and the following disclaimer;
2. distributions in binary form include the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other associated materials;
3. the copyright holder's name is not used to endorse products

built using this software without specific written permission.

DISCLAIMER

This software is provided 'as is' with no explicit or implied warranties in respect of any properties, including, but not limited to, correctness and fitness for purpose.

Portions contributed by Red Hat, including the pre-authentication plug-in framework and the NSS crypto implementation, contain the following copyright:

Copyright (C) 2006 Red Hat, Inc.
Portions copyright (C) 2006 Massachusetts Institute of Technology
All Rights Reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Red Hat, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The bundled verito source code is subject to the following license:

Copyright 2011 Red Hat, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation

files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

=====

The MS-KKDCP client implementation has the following copyright:

Copyright 2013,2014 Red Hat, Inc.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====

The implementations of GSSAPI mechglue in GSSAPI-SPNEGO in "src/lib/gssapi", including the following files:

```

lib/gssapi/generic/gssapi_err_generic.et
lib/gssapi/mechglue/g_accept_sec_context.c
lib/gssapi/mechglue/g_acquire_cred.c
lib/gssapi/mechglue/g_canon_name.c
lib/gssapi/mechglue/g_compare_name.c
lib/gssapi/mechglue/g_context_time.c
lib/gssapi/mechglue/g_delete_sec_context.c
lib/gssapi/mechglue/g_dsp_name.c
lib/gssapi/mechglue/g_dsp_status.c
lib/gssapi/mechglue/g_dup_name.c
lib/gssapi/mechglue/g_exp_sec_context.c
lib/gssapi/mechglue/g_export_name.c
lib/gssapi/mechglue/g_glue.c
lib/gssapi/mechglue/g_imp_name.c
lib/gssapi/mechglue/g_imp_sec_context.c
lib/gssapi/mechglue/g_init_sec_context.c
lib/gssapi/mechglue/g_initialize.c
lib/gssapi/mechglue/g_inquire_context.c
lib/gssapi/mechglue/g_inquire_cred.c
lib/gssapi/mechglue/g_inquire_names.c
lib/gssapi/mechglue/g_process_context.c
lib/gssapi/mechglue/g_rel_buffer.c
lib/gssapi/mechglue/g_rel_cred.c
lib/gssapi/mechglue/g_rel_name.c
lib/gssapi/mechglue/g_rel_oid_set.c
lib/gssapi/mechglue/g_seal.c
lib/gssapi/mechglue/g_sign.c
lib/gssapi/mechglue/g_store_cred.c
lib/gssapi/mechglue/g_unseal.c
lib/gssapi/mechglue/g_userok.c
lib/gssapi/mechglue/g_utils.c
lib/gssapi/mechglue/g_verify.c
lib/gssapi/mechglue/gssd_pname_to_uid.c
lib/gssapi/mechglue/mglueP.h
lib/gssapi/mechglue/oid_ops.c
lib/gssapi/spnego/gssapiP_spnego.h
lib/gssapi/spnego/spnego_mech.c

```

and the initial implementation of incremental propagation, including the following new or changed files:

```

include/ipropr_hdr.h
kadmin/server/ipropr_svc.c
lib/kdb/ipropr.x
lib/kdb/kdb_convert.c
lib/kdb/kdb_log.c
lib/kdb/kdb_log.h
lib/krb5/error_tables/kdb5_err.et
kprop/kpropd_rpc.c
kprop/kproplog.c

```

are subject to the following license:

Copyright (C) 2004 Sun Microsystems, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Kerberos V5 includes documentation and software developed at the University of California at Berkeley, which includes this copyright notice:

Copyright (C) 1983 Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions contributed by Novell, Inc., including the LDAP database backend, are subject to the following license:

Copyright (C) 2004-2005, Novell, Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * The copyright holder's name is not used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions funded by Sandia National Laboratory and developed by the University of Michigan's Center for Information Technology Integration, including the PKINIT implementation, are subject to the following license:

COPYRIGHT (C) 2006-2007
THE REGENTS OF THE UNIVERSITY OF MICHIGAN
ALL RIGHTS RESERVED

Permission is granted to use, copy, create derivative works and

redistribute this software and such derivative works for any purpose, so long as the name of The University of Michigan is not used in any advertising or publicity pertaining to the use of distribution of this software without specific, written prior authorization. If the above copyright notice or any other identification of the University of Michigan is included in any copy of any portion of this software, then the disclaimer below must also be included.

THIS SOFTWARE IS PROVIDED AS IS, WITHOUT REPRESENTATION FROM THE UNIVERSITY OF MICHIGAN AS TO ITS FITNESS FOR ANY PURPOSE, AND WITHOUT WARRANTY BY THE UNIVERSITY OF MICHIGAN OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE REGENTS OF THE UNIVERSITY OF MICHIGAN SHALL NOT BE LIABLE FOR ANY DAMAGES, INCLUDING SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WITH RESPECT TO ANY CLAIM ARISING OUT OF OR IN CONNECTION WITH THE USE OF THE SOFTWARE, EVEN IF IT HAS BEEN OR IS HEREAFTER ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

=====

The pkcs11.h file included in the PKINIT code has the following license:

Copyright 2006 g10 Code GmbH
Copyright 2006 Andreas Jellinghaus

This file is free software; as a special exception the author gives unlimited permission to copy and/or distribute it, with or without modifications, as long as this notice is preserved.

This file is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY, to the extent permitted by law; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

=====

Portions contributed by Apple Inc. are subject to the following license:

Copyright 2004-2008 Apple Inc. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that

the name of Apple Inc. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Apple Inc. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The implementations of UTF-8 string handling in src/util/support and src/lib/krb5/unicode are subject to the following copyright and permission notice:

The OpenLDAP Public License
Version 2.8, 17 August 2003

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions in source form must retain copyright statements and notices,
2. Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and
3. Redistributions must contain a verbatim copy of this document.

The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.

THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The names of the authors and copyright holders must not be used in

advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

=====

Marked test programs in src/lib/krb5/krb have the following copyright:

Copyright (C) 2006 Kungliga Tekniska Högskola
(Royal Institute of Technology, Stockholm, Sweden).
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of KTH nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY KTH AND ITS CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL KTH OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====

The KCM Mach RPC definition file used on macOS has the following copyright:

Copyright (C) 2009 Kungliga Tekniska Högskola
(Royal Institute of Technology, Stockholm, Sweden).
All rights reserved.

Portions Copyright (C) 2009 Apple Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above
copyright notice, this list of conditions and the following
disclaimer.
2. Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following
disclaimer in the documentation and/or other materials provided
with the distribution.
3. Neither the name of the Institute nor the names of its
contributors may be used to endorse or promote products derived
from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE INSTITUTE
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

=====

Portions of the RPC implementation in src/lib/rpc and
src/include/gssrpc have the following copyright and permission notice:

Copyright (C) 2010, Oracle America, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above
copyright notice, this list of conditions and the following
disclaimer.
2. Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following

disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the "Oracle America, Inc." nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====
Copyright (C) 2006,2007,2009 NTT (Nippon Telegraph and Telephone Corporation). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY NTT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL NTT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====
Copyright 2000 by Carnegie Mellon University

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Carnegie Mellon University not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

CARNEGIE MELLON UNIVERSITY DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL CARNEGIE MELLON UNIVERSITY BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (C) 2002 Naval Research Laboratory (NRL/CCS)

Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof.

NRL ALLOWS FREE USE OF THIS SOFTWARE IN ITS "AS IS" CONDITION AND DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.

Copyright (C) 1991, 1992, 1994 by Cygnus Support.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Cygnus Support makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Copyright (C) 2006 Secure Endpoints Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without

restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Portions of the implementation of the Fortuna-like PRNG are subject to the following notice:

Copyright (C) 2005 Marko Kreen
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (C) 1994 by the University of Southern California

EXPORT OF THIS SOFTWARE from the United States of America may

require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to copy, modify, and distribute this software and its documentation in source and binary forms is hereby granted, provided that any documentation or other materials related to such distribution or use acknowledge that the software was developed by the University of Southern California.

DISCLAIMER OF WARRANTY. THIS SOFTWARE IS PROVIDED "AS IS". The University of Southern California MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. By way of example, but not limitation, the University of Southern California MAKES NO REPRESENTATIONS OR WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. The University of Southern California shall not be held liable for any liability nor for any direct, indirect, or consequential damages with respect to any claim by the user or distributor of the ksu software.

Copyright (C) 1995
The President and Fellows of Harvard University

This code is derived from software contributed to Harvard by Jeremy Rassen.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the University of California, Berkeley and its contributors.
4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS

OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====
 Copyright (C) 2008 by the Massachusetts Institute of Technology.
 Copyright 1995 by Richard P. Basch. All Rights Reserved.
 Copyright 1995 by Lehman Brothers, Inc. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Richard P. Basch, Lehman Brothers and M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Richard P. Basch, Lehman Brothers and M.I.T. make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

=====
 The following notice applies to "src/lib/krb5/krb/strptime.c" and "src/include/k5-queue.h".

Copyright (C) 1997, 1998 The NetBSD Foundation, Inc.
 All rights reserved.

This code was contributed to The NetBSD Foundation by Klaus Klein.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the NetBSD Foundation, Inc. and its contributors.

4. Neither the name of The NetBSD Foundation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following notice applies to Unicode library files in "src/lib/krb5/unicode":

Copyright 1997, 1998, 1999 Computing Research Labs,
New Mexico State University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE COMPUTING RESEARCH LAB OR NEW MEXICO STATE UNIVERSITY BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following notice applies to "src/util/support/strncpy.c":

Copyright (C) 1998 Todd C. Miller "Todd.Miller@courtesan.com"

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

=====

The following notice applies to "src/util/profile/argv_parse.c" and "src/util/profile/argv_parse.h":

Copyright 1999 by Theodore Ts'o.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS" AND THEODORE TS'O (THE AUTHOR) DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. (Isn't it sick that the U.S. culture of lawsuit-happy lawyers requires this kind of disclaimer?)

=====

The following notice applies to SWIG-generated code in "src/util/profile/profile_tcl.c":

Copyright (C) 1999-2000, The University of Chicago

This file may be freely redistributed without license or fee provided this copyright message remains intact.

=====

The following notice applies to portions of "src/lib/rpc" and "src/include/gssrpc":

Copyright (C) 2000 The Regents of the University of Michigan. All rights reserved.

Copyright (C) 2000 Dug Song "dugsong@UMICH.EDU". All rights reserved, all wrongs reversed.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Implementations of the MD4 algorithm are subject to the following notice:

Copyright (C) 1990, RSA Data Security, Inc. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD4 Message Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD4 Message Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either

the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

=====

Implementations of the MD5 algorithm are subject to the following notice:

Copyright (C) 1990, RSA Data Security, Inc. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message- Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

=====

The following notice applies to
"src/lib/crypto/crypto_tests/t_md driver.c":

Copyright (C) 1990-2, RSA Data Security, Inc. Created 1990. All rights reserved.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

=====

Portions of "src/lib/krb5" are subject to the following notice:

Copyright (C) 1994 CyberSAFE Corporation.
Copyright 1990,1991,2007,2008 by the Massachusetts

Institute of Technology.
All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Furthermore if you modify this software you must label your software as modified software and not distribute it in such a fashion that it might be confused with the original M.I.T. software. Neither M.I.T., the Open Computing Security Group, nor CyberSAFE Corporation make any representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Portions contributed by PADL Software are subject to the following license:

Copyright (c) 2011, PADL Software Pty Ltd. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of PADL Software nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY PADL SOFTWARE AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL PADL SOFTWARE OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND

ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====

The bundled libev source code is subject to the following license:

All files in libev are Copyright (C)2007,2008,2009 Marc Alexander Lehmann.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Alternatively, the contents of this package may be used under the terms of the GNU General Public License ("GPL") version 2 or any later version, in which case the provisions of the GPL are applicable instead of the above. If you wish to allow the use of your version of this package only under the terms of the GPL and not to allow others to use your version of this file under the BSD license, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the GPL in this and the other files of this package. If you do not delete the provisions above, a recipient may use your version of this file under either the BSD or the GPL.

=====

Files copied from the Intel AESNI Sample Library are subject to the following license:

Copyright (C) 2010, Intel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Intel Corporation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following notice applies to
"src/ccapi/common/win/OldCC/autolock.hxx":

Copyright (C) 1998 by Danilo Almeida. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS

FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====

The following notice applies to portions of
 "src/plugins/preauth/spake/edwards25519.c" and
 "src/plugins/preauth/spake/edwards25519_tables.h":

The MIT License (MIT)

Copyright (c) 2015-2016 the fiat-crypto authors (see the AUTHORS file).

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

=====

The following notice applies to portions of
 "src/plugins/preauth/spake/edwards25519.c":

Copyright (c) 2015-2016, Google Inc.

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED

WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

1.59 less

1.59.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

Less License

Less
Copyright (C) 1984-2015 Mark Nudelman

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.60 libcap

1.60.1 Available under license

Unless otherwise **explicitly** stated, the following text describes the licensed conditions under which the contents of this libcap release may be used and distributed:

Redistribution and use in source and binary forms of libcap, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain any existing copyright notice, and this entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce all prior and current copyright notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of any author may not be used to endorse or promote products derived from this software without their specific prior written permission.

ALTERNATIVELY, this product may be distributed under the terms of the GNU General Public License (v2.0 - see below), in which case the provisions of the GNU GPL are required INSTEAD OF the above restrictions. (This clause is necessary due to a potential conflict between the GNU GPL and the restrictions contained in a BSD-style copyright.)

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR(S) BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Full text of gpl-2.0.txt:

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and

modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on

the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed

through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED

TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program

'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.61 libedit

1.61.1 Available under license

Copyright (c) 1992, 1993

The Regents of the University of California. All rights reserved.

This code is derived from software contributed to Berkeley by
Christos Zoulas of Cornell University.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

1.62 libestr

1.62.1 Available under license

libestr is available under the terms of the GNU LGPL v2.1 or above (full text below).

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use,
not price. Our General Public Licenses are designed to make sure that
you have the freedom to distribute copies of free software (and charge
for this service if you wish); that you receive source code or can get
it if you want it; that you can change the software and use pieces of
it in new free programs; and that you are informed that you can do
these things.

To protect your rights, we need to make restrictions that forbid
distributors to deny you these rights or to ask you to surrender these
rights. These restrictions translate to certain responsibilities for
you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them

with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in

non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact

all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under

the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative

work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with

this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free

programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

liblognorm, a fast samples-based log normalization library
Copyright (C) 2010 Rainer Gerhards

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

1.63 libevent

1.63.1 Available under license

Libevent is available for use under the following license, commonly known as the 3-clause (or "modified") BSD license:

```
=====
Copyright (c) 2000-2007 Niels Provos <provos@citi.umich.edu>
Copyright (c) 2007-2012 Niels Provos and Nick Mathewson
```

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```
=====
```

Portions of Libevent are based on works by others, also made available by them under the three-clause BSD license above. The copyright notices are available in the corresponding source files; the license is as above. Here's a list:

log.c:

Copyright (c) 2000 Dug Song <dugsong@monkey.org>
Copyright (c) 1993 The Regents of the University of California.

strlcpy.c:

Copyright (c) 1998 Todd C. Miller <Todd.Miller@courtesan.com>

win32select.c:

Copyright (c) 2003 Michael A. Davis <mike@datanerds.net>

evport.c:

Copyright (c) 2007 Sun Microsystems

ht-internal.h:

Copyright (c) 2002 Christopher Clark

minheap-internal.h:

Copyright (c) 2006 Maxim Yegorushkin <maxim.yegorushkin@gmail.com>

=====

The arc4module is available under the following, sometimes called the "OpenBSD" license:

Copyright (c) 1996, David Mazieres <dm@uun.org>

Copyright (c) 2008, Damien Miller <djm@openbsd.org>

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

1.64 libfastjson

1.64.1 Available under license

Copyright (c) 2015 Rainer Gerhards

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 2009-2012 Eric Haszlakiewicz

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 2004, 2005 Metaparadigm Pte Ltd

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.65 libffi

1.65.1 Available under license

libffi - Copyright (c) 1996-2014 Anthony Green, Red Hat, Inc and others.
See source files for details.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the ``Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ``AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.66 libgcc

1.66.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not

covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of

this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by

modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new

versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library.

It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

GCC RUNTIME LIBRARY EXCEPTION

Version 3.1, 31 March 2009

Copyright (C) 2009 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This GCC Runtime Library Exception ("Exception") is an additional permission under section 7 of the GNU General Public License, version 3 ("GPLv3"). It applies to a given file (the "Runtime Library") that bears a notice placed by the copyright holder of the file stating that the file is governed by GPLv3 along with this Exception.

When you use GCC to compile a program, GCC may combine portions of certain GCC header files and runtime libraries with the compiled program. The purpose of this Exception is to allow compilation of non-GPL (including proprietary) programs to use, in this way, the header files and runtime libraries covered by this Exception.

0. Definitions.

A file is an "Independent Module" if it either requires the Runtime Library for execution after a Compilation Process, or makes use of an interface provided by the Runtime Library, but is not otherwise based on the Runtime Library.

"GCC" means a version of the GNU Compiler Collection, with or without modifications, governed by version 3 (or a specified later version) of the GNU General Public License (GPL) with the option of using any subsequent versions published by the FSF.

"GPL-compatible Software" is software whose conditions of propagation, modification and use would permit combination with GCC in accord with the license of GCC.

"Target Code" refers to output from any compiler for a real or virtual target processor architecture, in executable form or suitable for input to an assembler, loader, linker and/or execution phase. Notwithstanding that, Target Code does not include data in any format that is used as a compiler intermediate representation, or used for producing a compiler intermediate representation.

The "Compilation Process" transforms code entirely represented in non-intermediate languages designed for human-written code, and/or in Java Virtual Machine byte code, into Target Code. Thus, for example, use of source code generators and preprocessors need not be considered part of the Compilation Process, since the Compilation Process can be understood as starting with the output of the generators or preprocessors.

A Compilation Process is "Eligible" if it is done using GCC, alone or with other GPL-compatible software, or if it is done without using any work based on GCC. For example, using non-GPL-compatible Software to optimize any GCC intermediate representations would not qualify as an Eligible Compilation Process.

1. Grant of Additional Permission.

You have permission to propagate a work of Target Code formed by combining the Runtime Library with Independent Modules, even if such propagation would otherwise violate the terms of GPLv3, provided that all Target Code was generated by Eligible Compilation Processes. You may then convey such a combination under terms of your choice, consistent with the licensing of the Independent Modules.

2. No Weakening of GCC Copyleft.

The availability of this Exception does not imply any general

presumption that third-party software is unaffected by the copyleft requirements of the license of GCC.

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer

can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the

extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its

content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the

written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply

if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free

patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM

IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it

is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

insert GPL v3 text here

GCC RUNTIME LIBRARY EXCEPTION

Version 3.1, 31 March 2009

General information:

<http://www.gnu.org/licenses/gcc-exception.html>

Copyright (C) 2009 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This GCC Runtime Library Exception ("Exception") is an additional permission under section 7 of the GNU General Public License, version 3 ("GPLv3"). It applies to a given file (the "Runtime Library") that bears a notice placed by the copyright holder of the file stating that the file is governed by GPLv3 along with this Exception.

When you use GCC to compile a program, GCC may combine portions of certain GCC header files and runtime libraries with the compiled program. The purpose of this Exception is to allow compilation of non-GPL (including proprietary) programs to use, in this way, the header files and runtime libraries covered by this Exception.

0. Definitions.

A file is an "Independent Module" if it either requires the Runtime Library for execution after a Compilation Process, or makes use of an interface provided by the Runtime Library, but is not otherwise based on the Runtime Library.

"GCC" means a version of the GNU Compiler Collection, with or without modifications, governed by version 3 (or a specified later version) of the GNU General Public License (GPL) with the option of using any subsequent versions published by the FSF.

"GPL-compatible Software" is software whose conditions of propagation, modification and use would permit combination with GCC in accord with the license of GCC.

"Target Code" refers to output from any compiler for a real or virtual target processor architecture, in executable form or suitable for input to an assembler, loader, linker and/or execution phase. Notwithstanding that, Target Code does not include data in any format that is used as a compiler intermediate representation, or used for producing a

compiler intermediate representation.

The "Compilation Process" transforms code entirely represented in non-intermediate languages designed for human-written code, and/or in Java Virtual Machine byte code, into Target Code. Thus, for example, use of source code generators and preprocessors need not be considered part of the Compilation Process, since the Compilation Process can be understood as starting with the output of the generators or preprocessors.

A Compilation Process is "Eligible" if it is done using GCC, alone or with other GPL-compatible software, or if it is done without using any work based on GCC. For example, using non-GPL-compatible Software to optimize any GCC intermediate representations would not qualify as an Eligible Compilation Process.

1. Grant of Additional Permission.

You have permission to propagate a work of Target Code formed by combining the Runtime Library with Independent Modules, even if such propagation would otherwise violate the terms of GPLv3, provided that all Target Code was generated by Eligible Compilation Processes. You may then convey such a combination under terms of your choice, consistent with the licensing of the Independent Modules.

2. No Weakening of GCC Copyleft.

The availability of this Exception does not imply any general presumption that third-party software is unaffected by the copyleft requirements of the license of GCC.

1.67 libgcrypt

1.67.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

^L

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

^L

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not

covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of

this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

^L

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

^L

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

^L

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by

modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

^L

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new

versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

^L

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

^L

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library.

It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.68 libgpg-error

1.68.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not

covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of

this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by

modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new

versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library.

It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

```
/* gpg-error.h - Public interface to libgpg-error.          -*- c -*-
 * Copyright (C) 2003, 2004, 2010, 2013, 2014, 2015, 2016 g10 Code GmbH
 *
 * This file is part of libgpg-error.
 *
 * libgpg-error is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public License
 * as published by the Free Software Foundation; either version 2.1 of
 * the License, or (at your option) any later version.
 *
 * libgpg-error is distributed in the hope that it will be useful, but
 * WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
```

```

* You should have received a copy of the GNU Lesser General Public
* License along with this program; if not, see <https://www.gnu.org/licenses/>.
*
* @configure_input@
*/

```

```

#ifndef GPG_ERROR_H
#define GPG_ERROR_H1

```

```

#include <stddef.h>
#include <stdio.h>
#include <stdarg.h>

```

```

/* The version string of this header. */
#define GPG_ERROR_VERSION @version@
#define GPGRT_VERSION @version@

```

```

/* The version number of this header. */
#define GPG_ERROR_VERSION_NUMBER @version-number@
#define GPGRT_VERSION_NUMBER @version-number@

```

```

#ifndef __GNUC__
# define GPG_ERR_INLINE __inline__
#elif defined(_MSC_VER) && _MSC_VER >= 1300
# define GPG_ERR_INLINE __inline
#elif defined(__STDC_VERSION__) && __STDC_VERSION__ >= 199901L
# define GPG_ERR_INLINE inline
#else
# ifndef GPG_ERR_INLINE
# define GPG_ERR_INLINE
# endif
#endif

```

```

#ifndef __cplusplus
extern "C" {
#if 0 /* just to make Emacs auto-indent happy */
}
#endif
#endif /* __cplusplus */

```

/* The GnuPG project consists of many components. Error codes are exchanged between all components. The common error codes and their user-presentable descriptions are kept into a shared library to allow adding new error codes and components without recompiling any of the other components. The interface will not change in a backward incompatible way.

An error code together with an error source build up an error value. As the error value is been passed from one component to another, it preserves the information about the source and nature of the error.

A component of the GnuPG project can define the following macros to tune the behaviour of the library:

GPG_ERR_SOURCE_DEFAULT: Define to an error source of type `gpg_err_source_t` to make that source the default for `gpg_error()`. Otherwise `GPG_ERR_SOURCE_UNKNOWN` is used as default.

GPG_ERR_ENABLE_GETTEXT_MACROS: Define to provide macros to map the internal gettext API to standard names. This has only an effect on Windows platforms.

GPGRT_ENABLE_ES_MACROS: Define to provide "es_" macros for the `estream` functions.

In addition to the error codes, Libgpg-error also provides a set of functions used by most GnuPG components. */

```
/* The error source type gpg_err_source_t.
```

```
    Where as the Poo out of a welle small
    Taketh his firste springing and his sours.
--Chaucer. */
```

```
/* Only use free slots, never change or reorder the existing
   entries. */
```

```
typedef enum
```

```
{
@include:err-sources@
/* This is one more than the largest allowed entry. */
    GPG_ERR_SOURCE_DIM = 128
} gpg_err_source_t;
```

```
/* The error code type gpg_err_code_t. */
```

```
/* Only use free slots, never change or reorder the existing
   entries. */
```

```
typedef enum
```

```
{
@include:err-codes@
/* The following error codes are used to map system errors. */
#define GPG_ERR_SYSTEM_ERROR(1 << 15)
@include:errnos@
/* This is one more than the largest allowed entry. */
    GPG_ERR_CODE_DIM = 65536
} gpg_err_code_t;
```

```
/* The error value type gpg_error_t. */
```

```
/* We would really like to use bit-fields in a struct, but using
   structs as return values can cause binary compatibility issues, in
   particular if you want to do it efficiently (also see
   -freg-struct-return option to GCC). */
```

```
typedef unsigned int gpg_error_t;
```

```

/* We use the lowest 16 bits of gpg_error_t for error codes. The 16th
   bit indicates system errors. */
#define GPG_ERR_CODE_MASK(GPG_ERR_CODE_DIM - 1)

/* Bits 17 to 24 are reserved. */

/* We use the upper 7 bits of gpg_error_t for error sources. */
#define GPG_ERR_SOURCE_MASK(GPG_ERR_SOURCE_DIM - 1)
#define GPG_ERR_SOURCE_SHIFT24

/* The highest bit is reserved. It shouldn't be used to prevent
   potential negative numbers when transmitting error values as
   text. */

/* GCC feature test. */
#if __GNUC__
# define _GPG_ERR_GCC_VERSION (__GNUC__ * 10000 \
                               + __GNUC_MINOR__ * 100 \
                               + __GNUC_PATCHLEVEL__)
#else
# define _GPG_ERR_GCC_VERSION 0
#endif

#undef _GPG_ERR_HAVE_CONSTRUCTOR
#if _GPG_ERR_GCC_VERSION > 30100
# define _GPG_ERR_CONSTRUCTOR __attribute__((__constructor__))
# define _GPG_ERR_HAVE_CONSTRUCTOR
#else
# define _GPG_ERR_CONSTRUCTOR
#endif

#define GPGRT_GCC_VERSION _GCC_ERR_GCC_VERSION

#if _GPG_ERR_GCC_VERSION >= 29200
# define _GPGRT__RESTRICT __restrict__
#else
# define _GPGRT__RESTRICT
#endif

/* The noreturn attribute. */
#if _GPG_ERR_GCC_VERSION >= 20500
# define GPGRT_ATTR_NORETURN __attribute__((noreturn))
#else
# define GPGRT_ATTR_NORETURN
#endif

/* The printf attributes. */
#if _GPG_ERR_GCC_VERSION >= 40400
# define GPGRT_ATTR_PRINTF(f, a) \
    __attribute__((format(__gnu_printf__, f, a)))
# define GPGRT_ATTR_NR_PRINTF(f, a) \
    __attribute__((noreturn, format(__gnu_printf__, f, a)))
#elif _GPG_ERR_GCC_VERSION >= 20500

```

```

# define GPGRT_ATTR_PRINTF(f, a) \
    __attribute__((format(printf,f,a)))
# define GPGRT_ATTR_NR_PRINTF(f, a) \
    __attribute__((noreturn, format(printf,f,a)))
#else
# define GPGRT_ATTR_PRINTF(f, a)
# define GPGRT_ATTR_NR_PRINTF(f, a)
#endif
#if _GPG_ERR_GCC_VERSION >= 20800
# define GPGRT_ATTR_FORMAT_ARG(a) __attribute__((__format_arg__(a)))
#else
# define GPGRT_ATTR_FORMAT_ARG(a)
#endif

/* The sentinel attribute. */
#if _GPG_ERR_GCC_VERSION >= 40000
# define GPGRT_ATTR_SENTINEL(a) __attribute__((sentinel(a)))
#else
# define GPGRT_ATTR_SENTINEL(a)
#endif

/* The used and unused attributes.
   I am not sure since when the unused attribute is really supported.
   In any case it is only needed for gcc versions which print a
   warning. Thus let us require gcc >= 3.5. */
#if _GPG_ERR_GCC_VERSION >= 40000
# define GPGRT_ATTR_USED __attribute__((used))
#else
# define GPGRT_ATTR_USED
#endif
#if _GPG_ERR_GCC_VERSION >= 30500
# define GPGRT_ATTR_UNUSED __attribute__((unused))
#else
# define GPGRT_ATTR_UNUSED
#endif

/* The deprecated attribute. */
#if _GPG_ERR_GCC_VERSION >= 30100
# define GPGRT_ATTR_DEPRECATED __attribute__((__deprecated__))
#else
# define GPGRT_ATTR_DEPRECATED
#endif

/* The pure attribute. */
#if _GPG_ERR_GCC_VERSION >= 29600
# define GPGRT_ATTR_PURE __attribute__((__pure__))
#else
# define GPGRT_ATTR_PURE
#endif

/* The malloc attribute. */
#if _GPG_ERR_GCC_VERSION >= 30200
# define GPGRT_ATTR_MALLOC __attribute__((__malloc__))
#else

```

```

# define GPGRT_ATTR_MALLOC
#endif

/* A macro defined if a GCC style __FUNCTION__ macro is available. */
#undef GPGRT_HAVE_MACRO_FUNCTION
#if _GPG_ERR_GCC_VERSION >= 20500
# define GPGRT_HAVE_MACRO_FUNCTION 1
#endif

/* A macro defined if the pragma GCC push_options is available. */
#undef GPGRT_HAVE_PRAGMA_GCC_PUSH
#if _GPG_ERR_GCC_VERSION >= 40400
# define GPGRT_HAVE_PRAGMA_GCC_PUSH 1
#endif

/* Detect LeakSanitizer (LSan) support for GCC and Clang based on
 * whether AddressSanitizer (ASAN) is enabled via -fsanitize=address).
 * Note that -fsanitize=leak just affect the linker options which
 * cannot be detected here. In that case you have to define the
 * GPGRT_HAVE_LEAK_SANITIZER macro manually. */
#ifdef __GNUC__
# ifdef __SANITIZE_ADDRESS__
# define GPGRT_HAVE_LEAK_SANITIZER
# elif defined(__has_feature)
# if __has_feature(address_sanitizer)
# define GPGRT_HAVE_LEAK_SANITIZER
# endif
# endif
#endif

/* The new name for the inline macro. */
#define GPGRT_INLINE GPG_ERR_INLINE

#ifdef GPGRT_HAVE_LEAK_SANITIZER
# include <sanitizer/lsan_interface.h>
#endif

/* Mark heap objects as non-leaked memory. */
static GPGRT_INLINE void
gpgmt_annotate_leaked_object (const void *p)
{
#ifdef GPGRT_HAVE_LEAK_SANITIZER
__lsan_ignore_object(p);
#else
(void)p;
#endif
}

/* Initialization function. */

/* Initialize the library. This function should be run early. */
gpg_error_t gpg_err_init (void) _GPG_ERR_CONSTRUCTOR;

```



```

/* If this is defined, the library is already initialized by the
   constructor and does not need to be initialized explicitly. */
#undef GPG_ERR_INITIALIZED
#ifdef _GPG_ERR_HAVE_CONSTRUCTOR
# define GPG_ERR_INITIALIZED1
# define gpg_err_init() do { gpg_err_init (); } while (0)
#else
# define gpg_err_init() do { ; } while (0)
#endif

/* See the source on how to use the deinit function; it is usually not
   required. */
void gpg_err_deinit (int mode);

/* Register blocking system I/O clamping functions. */
void gpg_err_set_syscall_clamp (void (*pre)(void), void (*post)(void));

/* Register a custom malloc/realloc/free function. */
void gpg_err_set_alloc_func (void *(*f)(void *, size_t n));

/* Constructor and accessor functions. */

/* Construct an error value from an error code and source. Within a
   subsystem, use gpg_error. */
static GPG_ERR_INLINE gpg_error_t
gpg_err_make (gpg_err_source_t source, gpg_err_code_t code)
{
    return code == GPG_ERR_NO_ERROR ? GPG_ERR_NO_ERROR
    : (((source & GPG_ERR_SOURCE_MASK) << GPG_ERR_SOURCE_SHIFT)
    | (code & GPG_ERR_CODE_MASK));
}

/* The user should define GPG_ERR_SOURCE_DEFAULT before including this
   file to specify a default source for gpg_error. */
#ifdef GPG_ERR_SOURCE_DEFAULT
#define GPG_ERR_SOURCE_DEFAULT GPG_ERR_SOURCE_UNKNOWN
#endif

static GPG_ERR_INLINE gpg_error_t
gpg_error (gpg_err_code_t code)
{
    return gpg_err_make (GPG_ERR_SOURCE_DEFAULT, code);
}

/* Retrieve the error code from an error value. */
static GPG_ERR_INLINE gpg_err_code_t
gpg_err_code (gpg_error_t err)
{
    return (gpg_err_code_t) (err & GPG_ERR_CODE_MASK);
}

```

```
/* Retrieve the error source from an error value. */
static GPG_ERR_INLINE gpg_err_source_t
gpg_err_source (gpg_error_t err)
{
    return (gpg_err_source_t) ((err >> GPG_ERR_SOURCE_SHIFT)
        & GPG_ERR_SOURCE_MASK);
}

/* String functions. */

/* Return a pointer to a string containing a description of the error
   code in the error value ERR. This function is not thread-safe. */
const char *gpg_strerror (gpg_error_t err);

/* Return the error string for ERR in the user-supplied buffer BUF of
   size BUFLen. This function is, in contrast to gpg_strerror,
   thread-safe if a thread-safe strerror_r() function is provided by
   the system. If the function succeeds, 0 is returned and BUF
   contains the string describing the error. If the buffer was not
   large enough, ERANGE is returned and BUF contains as much of the
   beginning of the error string as fits into the buffer. */
int gpg_strerror_r (gpg_error_t err, char *buf, size_t buflen);

/* Return a pointer to a string containing a description of the error
   source in the error value ERR. */
const char *gpg_strerror_source (gpg_error_t err);

/* Mapping of system errors (errno). */

/* Retrieve the error code for the system error ERR. This returns
   GPG_ERR_UNKNOWN_ERRNO if the system error is not mapped (report
   this). */
gpg_err_code_t gpg_err_code_from_errno (int err);

/* Retrieve the system error for the error code CODE. This returns 0
   if CODE is not a system error code. */
int gpg_err_code_to_errno (gpg_err_code_t code);

/* Retrieve the error code directly from the ERRNO variable. This
   returns GPG_ERR_UNKNOWN_ERRNO if the system error is not mapped
   (report this) and GPG_ERR_MISSING_ERRNO if ERRNO has the value 0. */
gpg_err_code_t gpg_err_code_from_syserror (void);

/* Set the ERRNO variable. This function is the preferred way to set
   ERRNO due to peculiarities on WindowsCE. */
void gpg_err_set_errno (int err);

/* Return or check the version. Both functions are identical. */
```

```
const char *gpgrt_check_version (const char *req_version);
const char *gpg_error_check_version (const char *req_version);

/* System specific type definitions. */
#define gpgrt_ssize_t
#define gpgrt_off_t

#include os-add

/* Self-documenting convenience functions. */

static GPG_ERR_INLINE gpg_error_t
gpg_err_make_from_errno (gpg_err_source_t source, int err)
{
    return gpg_err_make (source, gpg_err_code_from_errno (err));
}

static GPG_ERR_INLINE gpg_error_t
gpg_error_from_errno (int err)
{
    return gpg_error (gpg_err_code_from_errno (err));
}

static GPG_ERR_INLINE gpg_error_t
gpg_error_from_syserror (void)
{
    return gpg_error (gpg_err_code_from_syserror ());
}

/* Lock functions. */

#include lock-obj

#define GPGRT_LOCK_DEFINE(name) \
    static gpgrt_lock_t name = GPGRT_LOCK_INITIALIZER

/* NB: If GPGRT_LOCK_DEFINE is not used, zero out the lock variable
   before passing it to gpgrt_lock_init. */
gpg_err_code_t gpgrt_lock_init (gpgrt_lock_t *lockhd);
gpg_err_code_t gpgrt_lock_lock (gpgrt_lock_t *lockhd);
gpg_err_code_t gpgrt_lock_trylock (gpgrt_lock_t *lockhd);
gpg_err_code_t gpgrt_lock_unlock (gpgrt_lock_t *lockhd);
gpg_err_code_t gpgrt_lock_destroy (gpgrt_lock_t *lockhd);

/* Thread functions. */

gpg_err_code_t gpgrt_yield (void);
```

```

/* Estream */

/* The definition of this struct is entirely private. You must not
   use it for anything. It is only here so some functions can be
   implemented as macros. */
struct __gpgrt_stream_internal;
struct __gpgrt_stream
{
    /* The layout of this struct must never change. It may be grown,
       but only if all functions which access the new members are
       versioned. */

    /* Various flags. */
    struct {
        unsigned int magic: 16;
        unsigned int writing: 1;
        unsigned int reserved: 15;
    } flags;

    /* A pointer to the stream buffer. */
    unsigned char *buffer;

    /* The size of the buffer in bytes. */
    size_t buffer_size;

    /* The length of the usable data in the buffer, only valid when in
       read mode (see flags). */
    size_t data_len;

    /* The current position of the offset pointer, valid in read and
       write mode. */
    size_t data_offset;

    size_t data_flushed;
    unsigned char *unread_buffer;
    size_t unread_buffer_size;

    /* The number of unread bytes. */
    size_t unread_data_len;

    /* A pointer to our internal data for this stream. */
    struct __gpgrt_stream_internal *intern;
};

/* The opaque type for an estream. */
typedef struct __gpgrt_stream *gpgrt_stream_t;
#ifdef GPGRT_ENABLE_ES_MACROS
typedef struct __gpgrt_stream *estream_t;
#endif

typedef @api_ssize_t@ (*gpgrt_cookie_read_function_t) (void *cookie,
                                                         void *buffer, size_t size);
typedef @api_ssize_t@ (*gpgrt_cookie_write_function_t) (void *cookie,

```

```

        const void *buffer,
        size_t size);
typedef int (*gpgmt_cookie_seek_function_t) (void *cookie,
        gpgmt_off_t *pos, int whence);
typedef int (*gpgmt_cookie_close_function_t) (void *cookie);

struct _gpgmt_cookie_io_functions
{
    gpgmt_cookie_read_function_t func_read;
    gpgmt_cookie_write_function_t func_write;
    gpgmt_cookie_seek_function_t func_seek;
    gpgmt_cookie_close_function_t func_close;
};
typedef struct _gpgmt_cookie_io_functions gpgmt_cookie_io_functions_t;
#ifdef GPGMT_ENABLE_ES_MACROS
typedef struct _gpgmt_cookie_io_functions es_cookie_io_functions_t;
#define es_cookie_read_function_t gpgmt_cookie_read_function_t
#define es_cookie_write_function_t gpgmt_cookie_read_function_t
#define es_cookie_seek_function_t gpgmt_cookie_read_function_t
#define es_cookie_close_function_t gpgmt_cookie_read_function_t
#endif

enum gpgmt_syshd_types
{
    GPGMT_SYSHD_NONE = 0, /* No system handle available. */
    GPGMT_SYSHD_FD = 1, /* A file descriptor as returned by open(). */
    GPGMT_SYSHD_SOCKET = 2, /* A socket as returned by socket(). */
    GPGMT_SYSHD_RVID = 3, /* A rendezvous id (see libassuan's gpgcdev.c). */
    GPGMT_SYSHD_HANDLE = 4 /* A HANDLE object (Windows). */
};

struct _gpgmt_syshd
{
    enum gpgmt_syshd_types type;
    union {
        int fd;
        int sock;
        int rvid;
        void *handle;
    } u;
};
typedef struct _gpgmt_syshd gpgmt_syshd_t;
#ifdef GPGMT_ENABLE_ES_MACROS
typedef struct _gpgmt_syshd es_syshd_t;
#define ES_SYSHD_NONE GPGMT_SYSHD_NONE
#define ES_SYSHD_FD GPGMT_SYSHD_FD
#define ES_SYSHD_SOCKET GPGMT_SYSHD_SOCKET
#define ES_SYSHD_RVID GPGMT_SYSHD_RVID
#define ES_SYSHD_HANDLE GPGMT_SYSHD_HANDLE
#endif

/* The object used with gpgmt_poll. */
struct _gpgmt_poll_s
{

```

```

    gpgrt_stream_t stream;
    unsigned int want_read:1;
    unsigned int want_write:1;
    unsigned int want_oob:1;
    unsigned int want_rdhup:1;
    unsigned int _reserv1:4;
    unsigned int got_read:1;
    unsigned int got_write:1;
    unsigned int got_oob:1;
    unsigned int got_rdhup:1;
    unsigned int _reserv2:4;
    unsigned int got_err:1;
    unsigned int got_hup:1;
    unsigned int got_nval:1;
    unsigned int _reserv3:4;
    unsigned int ignore:1;
    unsigned int user:8;    /* For application use. */
};
typedef struct _gpgrt_poll_s gpgrt_poll_t;
#ifdef GPGRT_ENABLE_ES_MACROS
typedef struct _gpgrt_poll_s es_poll_t;
#endif

gpgrt_stream_t gpgrt_fopen (const char *_GPGRT__RESTRICT path,
                           const char *_GPGRT__RESTRICT mode);
gpgrt_stream_t gpgrt_mopen (void *_GPGRT__RESTRICT data,
                           size_t data_n, size_t data_len,
                           unsigned int grow,
                           void *(*func_realloc) (void *mem, size_t size),
                           void (*func_free) (void *mem),
                           const char *_GPGRT__RESTRICT mode);
gpgrt_stream_t gpgrt_fopenmem (size_t memlimit,
                              const char *_GPGRT__RESTRICT mode);
gpgrt_stream_t gpgrt_fopenmem_init (size_t memlimit,
                                    const char *_GPGRT__RESTRICT mode,
                                    const void *data, size_t datalen);
gpgrt_stream_t gpgrt_fdopen (int filedес, const char *mode);
gpgrt_stream_t gpgrt_fdopen_nc (int filedес, const char *mode);
gpgrt_stream_t gpgrt_sysopen (gpgrt_syshd_t *syshd, const char *mode);
gpgrt_stream_t gpgrt_sysopen_nc (gpgrt_syshd_t *syshd, const char *mode);
gpgrt_stream_t gpgrt_fopen (FILE *fp, const char *mode);
gpgrt_stream_t gpgrt_fopen_nc (FILE *fp, const char *mode);
gpgrt_stream_t gpgrt_freopen (const char *_GPGRT__RESTRICT path,
                              const char *_GPGRT__RESTRICT mode,
                              gpgrt_stream_t _GPGRT__RESTRICT stream);
gpgrt_stream_t gpgrt_fopencookie (void *_GPGRT__RESTRICT cookie,
                                  const char *_GPGRT__RESTRICT mode,
                                  gpgrt_cookie_io_functions_t functions);
int gpgrt_fclose (gpgrt_stream_t stream);
int gpgrt_fclose_snatch (gpgrt_stream_t stream,
                        void **r_buffer, size_t *r_bufllen);
int gpgrt_onclose (gpgrt_stream_t stream, int mode,
                  void (*fnc) (gpgrt_stream_t, void*), void *fnc_value);
int gpgrt_fileno (gpgrt_stream_t stream);
int gpgrt_fileno_unlocked (gpgrt_stream_t stream);

```

```

int gpgrt_syshd (gpgrt_stream_t stream, gpgrt_syshd_t *syshd);
int gpgrt_syshd_unlocked (gpgrt_stream_t stream, gpgrt_syshd_t *syshd);

void _gpgrt_set_std_fd (int no, int fd);
gpgrt_stream_t _gpgrt_get_std_stream (int fd);

#define gpgrt_stdin _gpgrt_get_std_stream (0)
#define gpgrt_stdout _gpgrt_get_std_stream (1)
#define gpgrt_stderr _gpgrt_get_std_stream (2)

void gpgrt_flockfile (gpgrt_stream_t stream);
int gpgrt_ftrylockfile (gpgrt_stream_t stream);
void gpgrt_funlockfile (gpgrt_stream_t stream);

int gpgrt_feof (gpgrt_stream_t stream);
int gpgrt_feof_unlocked (gpgrt_stream_t stream);
int gpgrt_ferror (gpgrt_stream_t stream);
int gpgrt_ferror_unlocked (gpgrt_stream_t stream);
void gpgrt_clearerr (gpgrt_stream_t stream);
void gpgrt_clearerr_unlocked (gpgrt_stream_t stream);

int _gpgrt_pending (gpgrt_stream_t stream); /* (private) */
int _gpgrt_pending_unlocked (gpgrt_stream_t stream); /* (private) */

#define gpgrt_pending(stream) _gpgrt_pending (stream)

#define gpgrt_pending_unlocked(stream)\
  (((!(stream)->flags.writing)\
    && (((stream)->data_offset < (stream)->data_len)\
      || ((stream)->unread_data_len)))\
    ? 1 : _gpgrt_pending_unlocked ((stream)))

int gpgrt_fflush (gpgrt_stream_t stream);
int gpgrt_fseek (gpgrt_stream_t stream, long int offset, int whence);
int gpgrt_fseeko (gpgrt_stream_t stream, gpgrt_off_t offset, int whence);
long int gpgrt_ftell (gpgrt_stream_t stream);
gpgrt_off_t gpgrt_ftello (gpgrt_stream_t stream);
void gpgrt_rewind (gpgrt_stream_t stream);

int gpgrt_fgetc (gpgrt_stream_t stream);
int gpgrt_fputc (int c, gpgrt_stream_t stream);

int _gpgrt_getc_underflow (gpgrt_stream_t stream); /* (private) */
int _gpgrt_putc_overflow (int c, gpgrt_stream_t stream); /* (private) */

#define gpgrt_getc_unlocked(stream)\
  (((!(stream)->flags.writing)\
    && ((stream)->data_offset < (stream)->data_len)\
    && (!(stream)->unread_data_len))\
    ? ((int) (stream)->buffer[((stream)->data_offset)++])\
    : _gpgrt_getc_underflow ((stream)))

#define gpgrt_putc_unlocked(c, stream)\

```

```

(((stream)->flags.writing\
 && ((stream)->data_offset < (stream)->buffer_size)\
 && (c != '\n'))\
 ? ((int) ((stream)->buffer[((stream)->data_offset)++] = (c)))\
 : _gpgrt_putc_overflow ((c), (stream)))

#define gpgrt_getc(stream)  gpgrt_fgetc (stream)
#define gpgrt_putc(c, stream) gpgrt_fputc (c, stream)

int gpgrt_ungetc (int c, gpgrt_stream_t stream);

int gpgrt_read (gpgrt_stream_t _GPGRT__RESTRICT stream,
                void * _GPGRT__RESTRICT buffer, size_t bytes_to_read,
                size_t * _GPGRT__RESTRICT bytes_read);
int gpgrt_write (gpgrt_stream_t _GPGRT__RESTRICT stream,
                const void * _GPGRT__RESTRICT buffer, size_t bytes_to_write,
                size_t * _GPGRT__RESTRICT bytes_written);
int gpgrt_write_sanitized (gpgrt_stream_t _GPGRT__RESTRICT stream,
                const void * _GPGRT__RESTRICT buffer, size_t length,
                const char *delimiters,
                size_t * _GPGRT__RESTRICT bytes_written);
int gpgrt_write_hexstring (gpgrt_stream_t _GPGRT__RESTRICT stream,
                const void * _GPGRT__RESTRICT buffer, size_t length,
                int reserved,
                size_t * _GPGRT__RESTRICT bytes_written);

size_t gpgrt_fread (void * _GPGRT__RESTRICT ptr, size_t size, size_t nitems,
                    gpgrt_stream_t _GPGRT__RESTRICT stream);
size_t gpgrt_fwrite (const void * _GPGRT__RESTRICT ptr, size_t size, size_t memb,
                    gpgrt_stream_t _GPGRT__RESTRICT stream);

char *gpgrt_fgets (char * _GPGRT__RESTRICT s, int n,
                    gpgrt_stream_t _GPGRT__RESTRICT stream);
int gpgrt_fputs (const char * _GPGRT__RESTRICT s,
                 gpgrt_stream_t _GPGRT__RESTRICT stream);
int gpgrt_fputs_unlocked (const char * _GPGRT__RESTRICT s,
                           gpgrt_stream_t _GPGRT__RESTRICT stream);

@api_ssize_t@ gpgrt_getline (char * _GPGRT__RESTRICT * _GPGRT__RESTRICT lineptr,
                             size_t * _GPGRT__RESTRICT n,
                             gpgrt_stream_t stream);
@api_ssize_t@ gpgrt_read_line (gpgrt_stream_t stream,
                               char **addr_of_buffer, size_t *length_of_buffer,
                               size_t *max_length);
void gpgrt_free (void *a);

int gpgrt_fprintf (gpgrt_stream_t _GPGRT__RESTRICT stream,
                  const char * _GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(2,3);
int gpgrt_fprintf_unlocked (gpgrt_stream_t _GPGRT__RESTRICT stream,
                            const char * _GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(2,3);

int gpgrt_printf (const char * _GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(1,2);

```



```

int gpgrt_printf_unlocked (const char *_GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(1,2);

int gpgrt_vfprintf (gpgrt_stream_t _GPGRT__RESTRICT stream,
    const char *_GPGRT__RESTRICT format, va_list ap)
    GPGRT_ATTR_PRINTF(2,0);
int gpgrt_vfprintf_unlocked (gpgrt_stream_t _GPGRT__RESTRICT stream,
    const char *_GPGRT__RESTRICT format, va_list ap)
    GPGRT_ATTR_PRINTF(2,0);

int gpgrt_setvbuf (gpgrt_stream_t _GPGRT__RESTRICT stream,
    char *_GPGRT__RESTRICT buf, int mode, size_t size);
void gpgrt_setbuf (gpgrt_stream_t _GPGRT__RESTRICT stream,
    char *_GPGRT__RESTRICT buf);

void gpgrt_set_binary (gpgrt_stream_t stream);
int gpgrt_set_nonblock (gpgrt_stream_t stream, int onoff);
int gpgrt_get_nonblock (gpgrt_stream_t stream);

int gpgrt_poll (gpgrt_poll_t *fdlist, unsigned int nfds, int timeout);

gpgrt_stream_t gpgrt_tmpfile (void);

void gpgrt_opaque_set (gpgrt_stream_t _GPGRT__RESTRICT stream,
    void *_GPGRT__RESTRICT opaque);
void *gpgrt_opaque_get (gpgrt_stream_t stream);

void gpgrt_fname_set (gpgrt_stream_t stream, const char *fname);
const char *gpgrt_fname_get (gpgrt_stream_t stream);

int gpgrt_asprintf (char **r_buf, const char *_GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(2,3);
int gpgrt_vasprintf (char **r_buf, const char *_GPGRT__RESTRICT format,
    va_list ap)
    GPGRT_ATTR_PRINTF(2,0);
char *gpgrt_bsprintf (const char *_GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(1,2);
char *gpgrt_vbsprintf (const char *_GPGRT__RESTRICT format, va_list ap)
    GPGRT_ATTR_PRINTF(1,0);
int gpgrt_snprintf (char *buf, size_t bufsz,
    const char *_GPGRT__RESTRICT format, ...)
    GPGRT_ATTR_PRINTF(3,4);
int gpgrt_vsnprintf (char *buf, size_t bufsz,
    const char *_GPGRT__RESTRICT format, va_list arg_ptr)
    GPGRT_ATTR_PRINTF(3,0);

#ifdef GPGRT_ENABLE_ES_MACROS
# define es_fopen      gpgrt_fopen
# define es_mopen      gpgrt_mopen
# define es_fopenmem    gpgrt_fopenmem
# define es_fopenmem_init gpgrt_fopenmem_init
# define es_fdopen      gpgrt_fdopen
# define es_fdopen_nc   gpgrt_fdopen_nc

```

```

# define es_sysopen      gpgrt_sysopen
# define es_sysopen_nc   gpgrt_sysopen_nc
# define es_fopen        gpgrt_fopen
# define es_fopen_nc     gpgrt_fopen_nc
# define es_freopen      gpgrt_freopen
# define es_fopencookie  gpgrt_fopencookie
# define es_fclose       gpgrt_fclose
# define es_fclose_snatch gpgrt_fclose_snatch
# define es_onclose      gpgrt_onclose
# define es_fileno       gpgrt_fileno
# define es_fileno_unlocked gpgrt_fileno_unlocked
# define es_syshd        gpgrt_syshd
# define es_syshd_unlocked gpgrt_syshd_unlocked
# define es_stdin         _gpgrt_get_std_stream(0)
# define es_stdout        _gpgrt_get_std_stream(1)
# define es_stderr        _gpgrt_get_std_stream(2)
# define es_flockfile    gpgrt_flockfile
# define es_ftrylockfile gpgrt_ftrylockfile
# define es_funlockfile  gpgrt_funlockfile
# define es_feof         gpgrt_feof
# define es_feof_unlocked gpgrt_feof_unlocked
# define es_ferror       gpgrt_ferror
# define es_ferror_unlocked gpgrt_ferror_unlocked
# define es_clearerr     gpgrt_clearerr
# define es_clearerr_unlocked gpgrt_clearerr_unlocked
# define es_pending      gpgrt_pending
# define es_pending_unlocked gpgrt_pending_unlocked
# define es_fflush       gpgrt_fflush
# define es_fseek        gpgrt_fseek
# define es_fseeko       gpgrt_fseeko
# define es_ftell        gpgrt_ftell
# define es_ftello       gpgrt_ftello
# define es_rewind       gpgrt_rewind
# define es_fgetc        gpgrt_fgetc
# define es_fputc        gpgrt_fputc
# define es_getc_unlocked gpgrt_getc_unlocked
# define es_putc_unlocked gpgrt_putc_unlocked
# define es_getc         gpgrt_getc
# define es_putc         gpgrt_putc
# define es_ungetc       gpgrt_ungetc
# define es_read         gpgrt_read
# define es_write        gpgrt_write
# define es_write_sanitized gpgrt_write_sanitized
# define es_write_hexstring gpgrt_write_hexstring
# define es_fread        gpgrt_fread
# define es_fwrite       gpgrt_fwrite
# define es_fgets        gpgrt_fgets
# define es_fputs        gpgrt_fputs
# define es_fputs_unlocked gpgrt_fputs_unlocked
# define es_getline      gpgrt_getline
# define es_read_line    gpgrt_read_line
# define es_free         gpgrt_free
# define es_fprintf      gpgrt_fprintf
# define es_fprintf_unlocked gpgrt_fprintf_unlocked
# define es_printf       gpgrt_printf

```

```

# define es_printf_unlocked  gpgrt_printf_unlocked
# define es_vfprintf         gpgrt_vfprintf
# define es_vfprintf_unlocked gpgrt_vfprintf_unlocked
# define es_setvbuf          gpgrt_setvbuf
# define es_setbuf           gpgrt_setbuf
# define es_set_binary       gpgrt_set_binary
# define es_set_nonblock     gpgrt_set_nonblock
# define es_get_nonblock     gpgrt_get_nonblock
# define es_poll             gpgrt_poll
# define es_tmpfile          gpgrt_tmpfile
# define es_opaque_set       gpgrt_opaque_set
# define es_opaque_get       gpgrt_opaque_get
# define es_fname_set        gpgrt_fname_set
# define es_fname_get        gpgrt_fname_get
# define es_asprintf         gpgrt_asprintf
# define es_vasprintf        gpgrt_vasprintf
# define es_bsprintf         gpgrt_bsprintf
# define es_vbsprintf        gpgrt_vbsprintf
#endif /*GPGRT_ENABLE_ES_MACROS*/

```

```

#ifdef __cplusplus

```

```

}

```

```

#endif

```

```

#endif /* GPG_ERROR_H */

```

```

/* init.c - Initialize the GnuPG error library.

```

```

   Copyright (C) 2005, 2010 g10 Code GmbH

```

This file is part of libpgp-error.

libpgp-error is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

libpgp-error is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this program; if not, see <<https://www.gnu.org/licenses/>>.

```

*/

```

```

#ifdef HAVE_CONFIG_H

```

```

#include <config.h>

```

```

#endif

```

```

#include <stdlib.h>

```

```

#include <stdio.h>

```

```

#include <string.h>

```

```

#include <errno.h>

```

```

#include "gpgrt-int.h"

```

```

#include "gettext.h"

```

```

#include "init.h"

#ifdef HAVE_W32CE_SYSTEM
# include "mkw32errmap.map.c" /* Generated map_w32codes () */
# ifndef TLS_OUT_OF_INDEXES
# define TLS_OUT_OF_INDEXES 0xFFFFFFFF
# endif
# ifndef __MINGW32CE__
# /* Replace the Mingw32CE provided abort function. */
# define abort() do { TerminateProcess (GetCurrentProcess(), 8); } while (0)
# endif
#endif

/* Locale directory support. */

#ifdef HAVE_W32_SYSTEM

#include <windows.h>

static int tls_index = TLS_OUT_OF_INDEXES; /* Index for the TLS functions. */

static char *get_locale_dir (void);
static void drop_locale_dir (char *locale_dir);

#else /*!HAVE_W32_SYSTEM*/

#define get_locale_dir() LOCALEDIR
#define drop_locale_dir(dir)

#endif /*!HAVE_W32_SYSTEM*/

/* The realloc function as set by gpgmt_set_alloc_func. */
static void *(*custom_realloc)(void *a, size_t n);

static void
real_init (void)
{
#ifdef ENABLE_NLS
    char *locale_dir;

    /* We only have to bind our locale directory to our text domain. */
    locale_dir = get_locale_dir ();
    if (locale_dir)
    {
        bindtextdomain (PACKAGE, locale_dir);
        drop_locale_dir (locale_dir);
    }
#endif
    _gpgmt_es_init ();
}

```

```

/* Initialize the library. This function should be run early. */
gpg_error_t
_gpg_err_init (void)
{
#ifdef HAVE_W32_SYSTEM
#ifdef DLL_EXPORT
/* We always have a constructor and thus this function is called
   automatically. Due to the way the C init code of mingw works,
   the constructors are called before ourDllMain function is
   called. The problem with that is that the TLS has not been setup
   and w32-gettext.c requires TLS. To solve this we do nothing here
   but call the actual init code from ourDllMain. */
#else /*!DLL_EXPORT*/
/* Note that if the TLS is actually used, we can't release the TLS
   as there is no way to know when a thread terminates (i.e. no
   thread-specific-atexit). You are really better off to use the
   DLL! */
if (tls_index == TLS_OUT_OF_INDEXES)
{
    tls_index = TlsAlloc ();
    if (tls_index == TLS_OUT_OF_INDEXES)
    {
        /* No way to continue - commit suicide. */
        abort ();
    }
    _gpg_w32__init_gettext_module ();
    real_init ();
}
#endif /*!DLL_EXPORT*/
#else
    real_init ();
#endif
    return 0;
}

/* Deinitialize libgpg-error. This function is only used in special
   circumstances. No gpg-error function should be used after this
   function has been called. A value of 0 passed for MODE
   deinitializes the entire libgpg-error, a value of 1 releases
   resources allocated for the current thread and only that thread may
   not anymore access libgpg-error after such a call. Under Windows
   this function may be called from theDllMain function of a DLL
   which statically links to libgpg-error. */
void
_gpg_err_deinit (int mode)
{
#ifdef HAVE_W32_SYSTEM && !defined(DLL_EXPORT)
    struct tls_space_s *tls;

    tls = TlsGetValue (tls_index);
    if (tls)
    {
        TlsSetValue (tls_index, NULL);
    }
#endif
}

```

```
        LocalFree (tls);
    }

    if (mode == 0)
    {
        TlsFree (tls_index);
        tls_index = TLS_OUT_OF_INDEXES;
    }
#else
    (void)mode;
#endif
}

/* Register F as allocation function. This function is used for all
   APIs which return an allocated buffer. F needs to have standard
   realloc semantics. It should be called as early as possible and
   not changed later. */
void
_gpgmt_set_alloc_func (void *(*f)(void *a, size_t n))
{
    custom_realloc = f;
}

/* The realloc to be used for data returned by the public API. */
void *
_gpgmt_realloc (void *a, size_t n)
{
    if (custom_realloc)
        return custom_realloc (a, n);

    if (!n)
    {
        free (a);
        return NULL;
    }

    if (!a)
        return malloc (n);

    return realloc (a, n);
}

/* The malloc to be used for data returned by the public API. */
void *
_gpgmt_malloc (size_t n)
{
    if (!n)
        n++;
    return _gpgmt_realloc (NULL, n);
}
```

```
/* The free to be used for data returned by the public API. */
void
_gpgmt_free (void *a)
{
    _gpgmt_realloc (a, 0);
}

void
_gpg_err_set_errno (int err)
{
#ifdef HAVE_W32CE_SYSTEM
    SetLastError (err);
#else /*!HAVE_W32CE_SYSTEM*/
    errno = err;
#endif /*!HAVE_W32CE_SYSTEM*/
}

#ifdef HAVE_W32_SYSTEM
/***** Below is only Windows code. *****/
/*****

static char *
get_locale_dir (void)
{
    static wchar_t moddir[MAX_PATH+5];
    char *result, *p;
    int nbytes;

    if (!GetModuleFileNameW (NULL, moddir, MAX_PATH))
        *moddir = 0;

#define SLDIR "\\share\\locale"
    if (*moddir)
    {
        nbytes = WideCharToMultiByte (CP_UTF8, 0, moddir, -1, NULL, 0, NULL, NULL);
        if (nbytes < 0)
            return NULL;

        result = malloc (nbytes + strlen (SLDIR) + 1);
        if (result)
        {
            nbytes = WideCharToMultiByte (CP_UTF8, 0, moddir, -1,
                                           result, nbytes, NULL, NULL);
            if (nbytes < 0)
            {
                free (result);
                result = NULL;
            }
        }
    }
}

```

```

else
{
    p = strrchr (result, '\\');
    if (p)
        *p = 0;
    /* If we are installed below "bin" strip that part and
       use the top directory instead.

       Background: Under Windows we don't install GnuPG
       below bin/ but in the top directory with only share/,
       lib/, and etc/ below it. One of the reasons is to
       keep the the length of the filenames at bay so not to
       increase the limited length of the PATH envvar.
       Another and more important reason, however, is that
       the very first GPG versions on W32 were installed
       into a flat directory structure and for best
       compatibility with these versions we didn't changed
       that later. For WindowsCE we can right away install
       it under bin, though. The hack with detection of the
       bin directory part allows us to eventually migrate to
       such a directory layout under plain Windows without
       the need to change libgpg-error. */
    p = strrchr (result, '\\');
    if (p && !strcmp (p+1, "bin"))
        *p = 0;
    /* Append the static part. */
    strcat (result, SLDIR);
}
}
else /* Use the old default value. */
{
    result = malloc (10 + strlen (SLDIR) + 1);
    if (result)
    {
        strcpy (result, "c:\\gnupg");
        strcat (result, SLDIR);
    }
}
#undef SLDIR
return result;
}

static void
drop_locale_dir (char *locale_dir)
{
    free (locale_dir);
}

/* Return the tls object. This function is guaranteed to return a
   valid non-NULL object. */
struct tls_space_s *
get_tls (void)

```



```

{
    struct tls_space_s *tls;

    tls = TlsGetValue (tls_index);
    if (!tls)
    {
        /* Called by a thread which existed before this DLL was loaded.
           Allocate the space. */
        tls = LocalAlloc (LPTR, sizeof *tls);
        if (!tls)
        {
            /* No way to continue - commit suicide. */
            abort ();
        }
        tls->gt_use_utf8 = 0;
        TlsSetValue (tls_index, tls);
    }

    return tls;
}

/* Return the value of the ERRNO variable. This needs to be a
   function so that we can have a per-thread ERRNO. This is used only
   on WindowsCE because that OS misses an errno. */
#ifdef HAVE_W32CE_SYSTEM
int
_gpg_w32ce_get_errno (void)
{
    return map_w32codes ( GetLastError () );
}
#endif /*HAVE_W32CE_SYSTEM*/

/* Replacement strerror function for WindowsCE. */
#ifdef HAVE_W32CE_SYSTEM
char *
_gpg_w32ce_strerror (int err)
{
    struct tls_space_s *tls = get_tls ();
    wchar_t tmpbuf[STRBUFFER_SIZE];
    int n;

    if (err == -1)
        err = _gpg_w32ce_get_errno ();

    /* Note: On a German HTC Touch Pro2 device I also tried
       LOCALE_USER_DEFAULT and LOCALE_SYSTEM_DEFAULT - both returned
       English messages. */
    if (FormatMessageW (FORMAT_MESSAGE_FROM_SYSTEM, NULL, err,
                       MAKELANGID (LANG_NEUTRAL, SUBLANG_DEFAULT),
                       tmpbuf, STRBUFFER_SIZE - 1,
                       NULL))
    {

```

```

        n = WideCharToMultiByte (CP_UTF8, 0, tmpbuf, -1,
                                tls->strerror_buffer,
                                sizeof tls->strerror_buffer -1,
                                NULL, NULL);
    }
    else
        n = -1;

    if (n < 0)
        snprintf (tls->strerror_buffer, sizeof tls->strerror_buffer -1,
                 "[w32err=%d]", err);
    return tls->strerror_buffer;
}
#endif /*HAVE_W32CE_SYSTEM*/

/* Entry point called by the DLL loader. */
#ifdef DLL_EXPORT
int WINAPI
DllMain (HINSTANCE hinst, DWORD reason, LPVOID reserved)
{
    struct tls_space_s *tls;
    (void)reserved;
    (void)hinst;

    switch (reason)
    {
        case DLL_PROCESS_ATTACH:
            tls_index = TlsAlloc ();
            if (tls_index == TLS_OUT_OF_INDEXES)
                return FALSE;
#ifdef _GPG_ERR_HAVE_CONSTRUCTOR
            /* If we have not constructors (e.g. MSC) we call it here. */
            _gpg_w32__init_gettext_module ();
#endif
            /* falltru. */
            case DLL_THREAD_ATTACH:
                tls = LocalAlloc (LPTR, sizeof *tls);
                if (!tls)
                    return FALSE;
                tls->gt_use_utf8 = 0;
                TlsSetValue (tls_index, tls);
                if (reason == DLL_PROCESS_ATTACH)
                {
                    real_init ();
                }
                break;

            case DLL_THREAD_DETACH:
                tls = TlsGetValue (tls_index);
                if (tls)
                    LocalFree (tls);
                break;

            case DLL_PROCESS_DETACH:

```

```
        tls = TlsGetValue (tls_index);
        if (tls)
            LocalFree (tls);
        TlsFree (tls_index);
        break;

    default:
        break;
}

return TRUE;
}
#endif /*DLL_EXPORT*/

#endif /*HAVE_W32_SYSTEM*/
```

1.69 libgssglue

1.69.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder say-

ing it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geograph-

ical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.70 libidn

1.70.1 Available under license

Libidn COPYING -- Explanation of licensing conditions.

Copyright (C) 2002-2015 Simon Josefsson

See the end for copying conditions.

The source code for the C library (libidn.a or libidn.so), the C# library (Libidn.dll) and the Java library (libidn-*.jar) are licensed under the terms of either the GNU General Public License version 2.0 or later (see the file COPYINGv2) or the GNU Lesser General Public License version 3.0 or later (see the file COPYING.LESSERv3), or both in parallel as here.

The author of the Java library has agreed to also distribute it under the Apache License Version 2.0 (see the file java/LICENSE-2.0).

The manual is licensed under the GNU Free Documentation License, Version 1.3 or any later.

The command line tool, self tests, examples, and other auxilliary files, are licensed under the GNU General Public License version 3.0 or later.

Other files are licensed as indicated in each file.

There may be exceptions to these general rules, see each file for precise information.

Copying and distribution of this file, with or without modification, are permitted in any medium without royalty provided the copyright notice and this notice are preserved.

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change

free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the

source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that

uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot

impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A

FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

GNU LESSER GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of

this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as

distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing

to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE

POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as

part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may

otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

```
/* idn.c --- Command line interface to libidn.
 * Copyright (C) 2003-2016 Simon Josefsson
 *
 * This file is part of GNU Libidn.
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */
```

```
#ifdef HAVE_CONFIG_H
# include "config.h"
#endif
```

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```



```

#include <errno.h>
#include <locale.h>
#include <unistd.h>

/* Gnulib headers. */
#include "error.h"
#include "gettext.h"
#define _(String) dgettext (PACKAGE, String)
#define N_(String) gettext_noop (String)
#include "progname.h"
#include "version-etc.h"

/* Libidn headers. */
#include <stringprep.h>
#include <punycode.h>
#include <idna.h>
#ifdef WITH_TLD
# include <tld.h>
#endif

#include "idn_cmd.h"

#define GREETING \
"Copyright 2002-2015 Simon Josefsson.\n" \
"GNU Libidn is free software with ABSOLUTELY NO WARRANTY. For more\n" \
"information about these matters, see <http://www.gnu.org/licenses/>.\n"

const char version_etc_copyright[] =
/* Do *not* mark this string for translation. %s is a copyright
   symbol suitable for this locale, and %d is the copyright
   year. */
"Copyright %s %d Simon Josefsson.";

static void
usage (int status)
{
  if (status != EXIT_SUCCESS)
    fprintf (stderr, _("Try '%s --help' for more information.\n"),
            program_name);
  else
    {
      printf (_("\n
Usage: %s [OPTION]... [STRINGS]...\n\
"), program_name);
      fputs (_("\n
Internationalized Domain Name (IDN) convert STRINGS, or standard input.\n\
\n\
"), stdout);
      fputs (_("\n
Command line interface to the internationalized domain name library.\n\
\n\
All strings are expected to be encoded in the preferred charset used\n\
by your locale. Use '--debug' to find out what this charset is. You\n\
can override the charset used by setting environment variable CHARSET.\n\

```

```

\n\
To process a string that starts with '-', for example '-foo', use '--'\n\
to signal the end of parameters, as in `idn --quiet -a -- -foo'.\n\
\n\
Mandatory arguments to long options are mandatory for short options too.\n\
"), stdout);
    fputs(_("
    -h, --help            Print help and exit\n\
    -V, --version        Print version and exit\n\
"), stdout);
    fputs(_("
    -s, --stringprep      Prepare string according to nameprep profile\n\
    -d, --punycode-decode  Decode Punycode\n\
    -e, --punycode-encode  Encode Punycode\n\
    -a, --idna-to-ascii    Convert to ACE according to IDNA (default mode)\n\
    -u, --idna-to-unicode  Convert from ACE according to IDNA\n\
"), stdout);
    fputs(_("
    --allow-unassigned    Toggle IDNA AllowUnassigned flag (default off)\n\
    --usestd3asciirules   Toggle IDNA UseSTD3ASCIIRules flag (default off)\n\
"), stdout);
    fputs(_("
    --no-tld              Don't check string for TLD specific rules\n\
                        Only for --idna-to-ascii and --idna-to-unicode\n\
"), stdout);
    fputs(_("
    -n, --nfkc            Normalize string according to Unicode v3.2 NFKC\n\
"), stdout);
    fputs(_("
    -p, --profile=STRING  Use specified stringprep profile instead\n\
                        Valid stringprep profiles: `Nameprep',\n\
                        `iSCSI', `Nodeprep', `Resourceprep',\n\
                        `trace', `SASLprep'\n\
"), stdout);
    fputs(_("
    --debug              Print debugging information\n\
    --quiet              Silent operation\n\
"), stdout);
    emit_bug_reporting_address();
}
exit(status);
}

int
main (int argc, char *argv[])
{
    struct gengetopt_args_info args_info;
    char *line = NULL;
    size_t linelen = 0;
    char *p, *r;
    uint32_t *q;
    unsigned cmdn = 0;
    int rc;

    setlocale (LC_ALL, "");

```

```

set_program_name (argv[0]);
bindtextdomain (PACKAGE, LOCALEDIR);
textdomain (PACKAGE);

if (cmdline_parser (argc, argv, &args_info) != 0)
    return EXIT_FAILURE;

if (args_info.version_given)
{
    version_etc (stdout, "idn", PACKAGE_NAME, VERSION,
"Simon Josefsson", (char *) NULL);
    return EXIT_SUCCESS;
}

if (args_info.help_given)
    usage (EXIT_SUCCESS);

/* Backwards compatibility: -n has always been the documented short
   form for --nfkc but, before v1.10, -k was the implemented short
   form. We now accept both to avoid documentation changes. */
if (args_info.hidden_nfkc_given)
    args_info.nfkc_given = 1;

if (!args_info.stringprep_given &&
    !args_info.punycode_encode_given && !args_info.punycode_decode_given &&
    !args_info.idna_to_ascii_given && !args_info.idna_to_unicode_given &&
    !args_info.nfkc_given)
    args_info.idna_to_ascii_given = 1;

if ((args_info.stringprep_given ? 1 : 0) +
    (args_info.punycode_encode_given ? 1 : 0) +
    (args_info.punycode_decode_given ? 1 : 0) +
    (args_info.idna_to_ascii_given ? 1 : 0) +
    (args_info.idna_to_unicode_given ? 1 : 0) +
    (args_info.nfkc_given ? 1 : 0) != 1)
{
    error (0, 0, "%s", _("only one of -s, -e, -d, -a, -u or -n can be specified"));
    usage (EXIT_FAILURE);
}

if (!args_info.quiet_given
    && args_info.inputs_num == 0
    && isatty (fileno (stdin)))
    fprintf (stderr, "%s %s\n" GREETING, PACKAGE, VERSION);

if (args_info.debug_given)
    fprintf (stderr, _("Charset '%s'.\n"), stringprep_locale_charset ());

if (!args_info.quiet_given
    && args_info.inputs_num == 0
    && isatty (fileno (stdin)))
    fprintf (stderr, "%s", _("Type each input string on a line by itself, "
        "terminated by a newline character.\n"));

```

```

do
{
    if (cmdn < args_info.inputs_num)
line = strdup (args_info.inputs[cmdn++]);
    else if (getline (&line, &linelen, stdin) == -1)
{
    if (feof (stdin))
        break;

    error (EXIT_FAILURE, errno, "%s", _("input error"));
}

    if (strlen (line) > 0)
if (line[strlen (line) - 1] == '\n')
    line[strlen (line) - 1] = '\0';

    if (args_info.stringprep_given)
{
p = stringprep_locale_to_utf8 (line);
if (!p)
    error (EXIT_FAILURE, 0, _("could not convert from %s to UTF-8"),
stringprep_locale_charset ());

q = stringprep_utf8_to_ucs4 (p, -1, NULL);
if (!q)
{
    free (p);
    error (EXIT_FAILURE, 0, "%s",
_("could not convert from UTF-8 to UCS-4"));
}

if (args_info.debug_given)
{
    size_t i;
    for (i = 0; q[i]; i++)
fprintf (stderr, "input[%lu] = U+%04x\n",
(unsigned long) i, q[i]);
}
free (q);

rc = stringprep_profile (p, &r,
args_info.profile_given ?
args_info.profile_arg : "Nameprep", 0);
free (p);
if (rc != STRINGPREP_OK)
    error (EXIT_FAILURE, 0, _("stringprep_profile: %s"),
stringprep_strerror (rc));

q = stringprep_utf8_to_ucs4 (r, -1, NULL);
if (!q)
{
    free (r);
    error (EXIT_FAILURE, 0, "%s",
_("could not convert from UTF-8 to UCS-4"));
}

```

```
if (args_info.debug_given)
{
    size_t i;
    for (i = 0; q[i]; i++)
fprintf (stderr, "output[%lu] = U+%04x\n",
(unsigned long) i, q[i]);
}
free (q);

p = stringprep_utf8_to_locale (r);
free (r);
if (!p)
    error (EXIT_FAILURE, 0, _("could not convert from UTF-8 to %s"),
stringprep_locale_charset ());

fprintf (stdout, "%s\n", p);

free (p);
}

if (args_info.punycode_encode_given)
{
    char encbuf[BUFSIZ];
    size_t len, len2;

    p = stringprep_locale_to_utf8 (line);
    if (!p)
        error (EXIT_FAILURE, 0, _("could not convert from %s to UTF-8"),
stringprep_locale_charset ());

    q = stringprep_utf8_to_ucs4 (p, -1, &len);
    free (p);
    if (!q)
        error (EXIT_FAILURE, 0, "%s",
_("could not convert from UTF-8 to UCS-4"));

    if (args_info.debug_given)
    {
        size_t i;
        for (i = 0; i < len; i++)
fprintf (stderr, "input[%lu] = U+%04x\n",
(unsigned long) i, q[i]);
    }

    len2 = BUFSIZ - 1;
    rc = punycode_encode (len, q, NULL, &len2, encbuf);
    free (q);
    if (rc != PUNYCODE_SUCCESS)
        error (EXIT_FAILURE, 0, _("punycode_encode: %s"),
punycode_strerror (rc));

    encbuf[len2] = '\0';
```

```

p = stringprep_utf8_to_locale (encbuf);
if (!p)
    error (EXIT_FAILURE, 0, _("could not convert from UTF-8 to %s"),
    stringprep_locale_charset ());

fprintf (stdout, "%s\n", p);

free (p);
}

if (args_info.punycode_decode_given)
{
    size_t len;

    len = BUFSIZ;
    q = (uint32_t *) malloc (len * sizeof (q[0]));
    if (!q)
        error (EXIT_FAILURE, ENOMEM, N_("malloc"));

    rc = punycode_decode (strlen (line), line, &len, q, NULL);
    if (rc != PUNYCODE_SUCCESS)
    {
        free (q);
        error (EXIT_FAILURE, 0, _("punycode_decode: %s"),
        punycode_strerror (rc));
    }

    if (args_info.debug_given)
    {
        size_t i;
        for (i = 0; i < len; i++)
            fprintf (stderr, "output[%lu] = U+%04x\n",
            (unsigned long) i, q[i]);
    }

    q[len] = 0;
    r = stringprep_ucs4_to_utf8 (q, -1, NULL, NULL);
    free (q);
    if (!r)
        error (EXIT_FAILURE, 0, "%s",
        _("could not convert from UCS-4 to UTF-8"));

    p = stringprep_utf8_to_locale (r);
    free (r);
    if (!r)
        error (EXIT_FAILURE, 0, _("could not convert from UTF-8 to %s"),
        stringprep_locale_charset ());

    fprintf (stdout, "%s\n", p);

    free (p);
}

if (args_info.idna_to_ascii_given)
{

```

```

p = stringprep_locale_to_utf8 (line);
if (!p)
    error (EXIT_FAILURE, 0, _("could not convert from %s to UTF-8"),
    stringprep_locale_charset ());

q = stringprep_utf8_to_ucs4 (p, -1, NULL);
free (p);
if (!q)
    error (EXIT_FAILURE, 0, "%s",
    _("could not convert from UCS-4 to UTF-8"));

if (args_info.debug_given)
{
    size_t i;
    for (i = 0; q[i]; i++)
        fprintf (stderr, "input[%lu] = U+%04x\n",
        (unsigned long) i, q[i]);
}

rc = idna_to_ascii_4z (q, &p,
    (args_info.allow_unassigned_given ?
    IDNA_ALLOW_UNASSIGNED : 0) |
    (args_info.usestd3asciirules_given ?
    IDNA_USE_STD3_ASCII_RULES : 0));
free (q);
if (rc != IDNA_SUCCESS)
    error (EXIT_FAILURE, 0, _("idna_to_ascii_4z: %s"),
    idna_strerror (rc));

#ifdef WITH_TLD
if (args_info.tld_flag && !args_info.no_tld_flag)
{
    size_t errpos;

    rc = idna_to_unicode_8z4z (p, &q,
    (args_info.allow_unassigned_given ?
    IDNA_ALLOW_UNASSIGNED : 0) |
    (args_info.usestd3asciirules_given ?
    IDNA_USE_STD3_ASCII_RULES : 0));
    if (rc != IDNA_SUCCESS)
        error (EXIT_FAILURE, 0, _("idna_to_unicode_8z4z (TLD): %s"),
        idna_strerror (rc));

    if (args_info.debug_given)
    {
        size_t i;
        for (i = 0; q[i]; i++)
            fprintf (stderr, "tld[%lu] = U+%04x\n",
            (unsigned long) i, q[i]);
    }

    rc = tld_check_4z (q, &errpos, NULL);
    free (q);
    if (rc == TLD_INVALID)

```

```

error (EXIT_FAILURE, 0, _("tld_check_4z (position %lu): %s"),
      (unsigned long) errpos, tld_strerror (rc));
if (rc != TLD_SUCCESS)
error (EXIT_FAILURE, 0, _("tld_check_4z: %s"),
      tld_strerror (rc));
}
#endif

if (args_info.debug_given)
{
    size_t i;
    for (i = 0; p[i]; i++)
fprintf (stderr, "output[%lu] = U+%04x\n",
        (unsigned long) i, p[i]);
}

fprintf (stdout, "%s\n", p);

free (p);
}

if (args_info.idna_to_unicode_given)
{
    p = stringprep_locale_to_utf8 (line);
    if (!p)
        error (EXIT_FAILURE, 0, _("could not convert from %s to UTF-8"),
              stringprep_locale_charset ());

    q = stringprep_utf8_to_ucs4 (p, -1, NULL);
    if (!q)
    {
        free (p);
        error (EXIT_FAILURE, 0, "%s",
              _("could not convert from UCS-4 to UTF-8"));
    }

    if (args_info.debug_given)
    {
        size_t i;
        for (i = 0; q[i]; i++)
fprintf (stderr, "input[%lu] = U+%04x\n",
        (unsigned long) i, q[i]);
    }
    free (q);

    rc = idna_to_unicode_8z4z (p, &q,
        (args_info.allow_unassigned_given ?
         IDNA_ALLOW_UNASSIGNED : 0) |
        (args_info.usestd3asciirules_given ?
         IDNA_USE_STD3_ASCII_RULES : 0));
    free (p);
    if (rc != IDNA_SUCCESS)
        error (EXIT_FAILURE, 0, _("idna_to_unicode_8z4z: %s"),
              idna_strerror (rc));
}

```



```

    if (args_info.debug_given)
    {
        size_t i;
        for (i = 0; q[i]; i++)
            fprintf(stderr, "output[%lu] = U+%04x\n",
                (unsigned long) i, q[i]);
    }

#ifdef WITH_TLD
    if (args_info.tld_flag)
    {
        size_t errpos;

        rc = tld_check_4z (q, &errpos, NULL);
        if (rc == TLD_INVALID)
        {
            free (q);
            error (EXIT_FAILURE, 0,
                _("tld_check_4z (position %lu): %s"),
                (unsigned long) errpos, tld_strerror (rc));
        }
        if (rc != TLD_SUCCESS)
        {
            free (q);
            error (EXIT_FAILURE, 0, _("tld_check_4z: %s"),
                tld_strerror (rc));
        }
    }
#endif

    r = stringprep_ucs4_to_utf8 (q, -1, NULL, NULL);
    free (q);
    if (!r)
        error (EXIT_FAILURE, 0, "%s",
            _("could not convert from UTF-8 to UCS-4"));

    p = stringprep_utf8_to_locale (r);
    free (r);
    if (!p)
        error (EXIT_FAILURE, 0, _("could not convert from UTF-8 to %s"),
            stringprep_locale_charset ());

    fprintf (stdout, "%s\n", p);

    free (p);
}

    if (args_info.nfkc_given)
    {
        p = stringprep_locale_to_utf8 (line);
        if (!p)
            error (EXIT_FAILURE, 0, _("could not convert from %s to UTF-8"),
                stringprep_locale_charset ());
    }

```

```

    if (args_info.debug_given)
    {
        size_t i;

        q = stringprep_utf8_to_ucs4 (p, -1, NULL);
        if (!q)
        {
            free (p);
            error (EXIT_FAILURE, 0, "%s",
                _("could not convert from UTF-8 to UCS-4"));
        }

        for (i = 0; q[i]; i++)
            fprintf (stderr, "input[%lu] = U+%04x\n",
                (unsigned long) i, q[i]);

        free (q);
    }

    r = stringprep_utf8_nfkc_normalize (p, -1);
    free (p);
    if (!r)
        error (EXIT_FAILURE, 0, "%s",
            _("could not do NFKC normalization"));

    if (args_info.debug_given)
    {
        size_t i;

        q = stringprep_utf8_to_ucs4 (r, -1, NULL);
        if (!q)
        {
            free (r);
            error (EXIT_FAILURE, 0, "%s",
                _("could not convert from UTF-8 to UCS-4"));
        }

        for (i = 0; q[i]; i++)
            fprintf (stderr, "output[%lu] = U+%04x\n",
                (unsigned long) i, q[i]);

        free (q);
    }

    p = stringprep_utf8_to_locale (r);
    free (r);
    if (!p)
        error (EXIT_FAILURE, 0, _("could not convert from UTF-8 to %s"),
            stringprep_locale_charset ());

    fprintf (stdout, "%s\n", p);

    free (p);
}

```

```

        fflush (stdout);
    }
    while (!feof (stdin) && !ferror (stdin) && (args_info.inputs_num == 0 ||
        cmdn < args_info.inputs_num));

    free (line);

    return EXIT_SUCCESS;
}
/* idna.h --- Prototypes for Internationalized Domain Name library.
   Copyright (C) 2002-2016 Simon Josefsson

```

This file is part of GNU Libidn.

GNU Libidn is free software: you can redistribute it and/or modify it under the terms of either:

- * the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

or

- * the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

or both in parallel, as here.

GNU Libidn is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received copies of the GNU General Public License and the GNU Lesser General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>. */

```

#ifndef IDNA_H
#define IDNA_H

#ifndef IDNAPI
# if defined LIBIDN_BUILDING && defined HAVE_VISIBILITY && HAVE_VISIBILITY
#  define IDNAPI __attribute__((visibility__("default")))
# elif defined LIBIDN_BUILDING && defined _MSC_VER && !defined LIBIDN_STATIC
#  define IDNAPI __declspec(dllexport)
# elif defined _MSC_VER && !defined LIBIDN_STATIC
#  define IDNAPI __declspec(dllimport)
# else
#  define IDNAPI
# endif
#endif

#include <stddef.h> /* size_t */

```

```

#include <idn-int.h> /* uint32_t */

#ifdef __cplusplus
extern "C"
{
#endif

/* Error codes. */
typedef enum
{
    IDNA_SUCCESS = 0,
    IDNA_STRINGPREP_ERROR = 1,
    IDNA_PUNYCODE_ERROR = 2,
    IDNA_CONTAINS_NON_LDH = 3,
    /* Workaround typo in earlier versions. */
    IDNA_CONTAINS_LDH = IDNA_CONTAINS_NON_LDH,
    IDNA_CONTAINS_MINUS = 4,
    IDNA_INVALID_LENGTH = 5,
    IDNA_NO_ACE_PREFIX = 6,
    IDNA_ROUNDTRIP_VERIFY_ERROR = 7,
    IDNA_CONTAINS_ACE_PREFIX = 8,
    IDNA_ICONV_ERROR = 9,
    /* Internal errors. */
    IDNA_MALLOC_ERROR = 201,
    IDNA_DLOPEN_ERROR = 202
} Idna_rc;

/* IDNA flags */
typedef enum
{
    IDNA_ALLOW_UNASSIGNED = 0x0001,
    IDNA_USE_STD3_ASCII_RULES = 0x0002
} Idna_flags;

#ifdef IDNA_ACE_PREFIX
# define IDNA_ACE_PREFIX "xn--"
#endif

extern IDNAPI const char *idna_strerror (Idna_rc rc);

/* Core functions */
extern IDNAPI int idna_to_ascii_4i (const uint32_t * in, size_t inlen,
    char *out, int flags);
extern IDNAPI int idna_to_unicode_44i (const uint32_t * in, size_t inlen,
    uint32_t * out, size_t * outlen,
    int flags);

/* Wrappers that handle several labels */

extern IDNAPI int idna_to_ascii_4z (const uint32_t * input,
    char **output, int flags);

extern IDNAPI int idna_to_ascii_8z (const char *input, char **output,
    int flags);

```

```
extern IDNAPI int idna_to_ascii_lz (const char *input, char **output,
    int flags);

extern IDNAPI int idna_to_unicode_4z4z (const uint32_t * input,
    uint32_t ** output, int flags);

extern IDNAPI int idna_to_unicode_8z4z (const char *input,
    uint32_t ** output, int flags);

extern IDNAPI int idna_to_unicode_8z8z (const char *input,
    char **output, int flags);

extern IDNAPI int idna_to_unicode_8z1z (const char *input,
    char **output, int flags);

extern IDNAPI int idna_to_unicode_lzlz (const char *input,
    char **output, int flags);

#ifdef __cplusplus
}
#endif

#endif/* IDNA_H */
```

1.71 libnewt

1.71.1 Available under license

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some
specially designated Free Software Foundation software, and to any
other libraries whose authors decide to use it. You can use it for
your libraries, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid
anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if
you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link a program with the library, you must provide
complete object files to the recipients so that they can relink them
with the library, after making changes to the library and recompiling
it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright
the library, and (2) offer you this license which gives you legal
permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain

that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of

the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by

this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library

facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed

through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR

CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.72 libnfsidmap

1.72.1 Available under license

Copyright (c) 2004 The Regents of the University of Michigan.
All rights reserved.

Marius Aamodt Eriksen <marius@umich.edu>
J. Bruce Fields <bfields@umich.edu>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.73 libpam

1.73.1 Available under license

Unless otherwise **explicitly** stated the following text describes the licensed conditions under which the contents of this Linux-PAM release may be distributed:

Redistribution and use in source and binary forms of Linux-PAM, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain any existing copyright notice, and this entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce all prior and current copyright notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of any author may not be used to endorse or promote products derived from this software without their specific prior written permission.

ALTERNATIVELY, this product may be distributed under the terms of the GNU General Public License, in which case the provisions of the GNU GPL are required **INSTEAD OF** the above restrictions. (This clause is necessary due to a potential conflict between the GNU GPL and the restrictions contained in a BSD-style copyright.)

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR(S) BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.74 libpcap

1.74.1 Available under license

License: BSD

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

/*

* Copyright (c) 1993, 1994, 1995, 1996, 1997

*The Regents of the University of California. All rights reserved.

*

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:

- * 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
- * 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
- * 3. All advertising materials mentioning features or use of this software
* must display the following acknowledgement:

*This product includes software developed by the Computer Systems

*Engineering Group at Lawrence Berkeley Laboratory.

- * 4. Neither the name of the University nor of the Laboratory may be used
* to endorse or promote products derived from this software without
* specific prior written permission.

*

* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT

```
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*/
```

```
/*
* For backwards compatibility.
*
* Note to OS vendors: do NOT get rid of this file! Many applications
* expect to be able to include <pcap.h>, and at least some of them
* go through contortions in their configure scripts to try to detect
* OSes that have "helpfully" moved pcap.h to <pcap/pcap.h> without
* leaving behind a <pcap.h> file.
*/
#include <pcap/pcap.h>
```

1.75 libpcre

1.75.1 Available under license

PCRE LICENCE

PCRE is a library of functions to support regular expressions whose syntax and semantics are as close as possible to those of the Perl 5 language.

Release 8 of PCRE is distributed under the terms of the "BSD" licence, as specified below. The documentation for PCRE, supplied in the "doc" directory, is distributed under the same terms as the software itself. The data in the testdata directory is not copyrighted and is in the public domain.

The basic library functions are written in C and are freestanding. Also included in the distribution is a set of C++ wrapper functions, and a just-in-time compiler that can be used to optimize pattern matching. These are both optional features that can be omitted when the library is built.

THE BASIC LIBRARY FUNCTIONS

Written by: Philip Hazel
Email local part: ph10
Email domain: cam.ac.uk

University of Cambridge Computing Service,
Cambridge, England.

Copyright (c) 1997-2016 University of Cambridge
All rights reserved.

PCRE JUST-IN-TIME COMPILATION SUPPORT

Written by: Zoltan Herczeg
Email local part: hzmester
Email domain: freemail.hu

Copyright(c) 2010-2016 Zoltan Herczeg
All rights reserved.

STACK-LESS JUST-IN-TIME COMPILER

Written by: Zoltan Herczeg
Email local part: hzmester

Email domain: freemail.hu

Copyright(c) 2009-2016 Zoltan Herczeg
All rights reserved.

THE C++ WRAPPER FUNCTIONS

Contributed by: Google Inc.

Copyright (c) 2007-2012, Google Inc.
All rights reserved.

THE "BSD" LICENCE

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
- * Neither the name of the University of Cambridge nor the name of Google
Inc. nor the names of their contributors may be used to endorse or
promote products derived from this software without specific prior
written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

End

1.76 libproxy

1.76.1 Available under license

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use,
not price. Our General Public Licenses are designed to make sure that
you have the freedom to distribute copies of free software (and charge
for this service if you wish); that you receive source code or can get
it if you want it; that you can change the software and use pieces of
it in new free programs; and that you are informed that you can do
these things.

To protect your rights, we need to make restrictions that forbid
distributors to deny you these rights or to ask you to surrender these
rights. These restrictions translate to certain responsibilities for
you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them
with the library after making changes to the library and recompiling
it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public

License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception,

the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent

infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free

Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

/*****

* libproxy - A library for proxy configuration
* Copyright (C) 2006 Nathaniel McCallum <nathaniel@natemccallum.com>
*

* This library is free software; you can redistribute it and/or
* modify it under the terms of the GNU Lesser General Public
* License as published by the Free Software Foundation; either
* version 2.1 of the License, or (at your option) any later version.
*

* This library is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
* Lesser General Public License for more details.
*

* You should have received a copy of the GNU Lesser General Public
* License along with this library; if not, write to the Free Software
* Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
*****/

#define _BSD_SOURCE

#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include <string.h>

/* Import libproxy API */
#include <proxy.h>

```
void *
malloc0(size_t s)
{
    void *tmp = malloc(s);
    if (!tmp) return NULL;
    memset(tmp, '0', s);
    return tmp;
}
```

```
/**
 * Prints an array of proxies. Proxies are space separated.
 * @proxies an array containing the proxies returned by libproxy.
 */
void
print_proxies(char **proxies)
{
    int j;

    if (!proxies) {
        printf("\n");
        return;
    }

    for (j=0; proxies[j] ; j++)
    {
        printf("%s%s", proxies[j], proxies[j+1] ? " " : "\n");
        free(proxies[j]);
    }
    free(proxies);
}

int
main(int argc, char **argv)
{
    int i;
    char url[102400]; // Should be plenty long for most URLs

    /* Create the proxy factory object */
    pxProxyFactory *pf = px_proxy_factory_new();
    if (!pf)
    {
        fprintf(stderr, "An unknown error occurred!\n");
        return 1;
    }
    /* User entered some arguments on startup. skip interactive */
    if (argc > 1)
    {
        for (i=1; i < argc ; i++)
        {
            /*
             * Get an array of proxies to use. These should be used
             * in the order returned. Only move on to the next proxy
             * if the first one fails (etc).
             */
            print_proxies(px_proxy_factory_get_proxies(pf, argv[i]));
        }
    }
    /* Interactive mode */
    else
    {
        /* For each URL we read on STDIN, get the proxies to use */
        for (url[0] = '\0' ; fgets(url, 102400, stdin) != NULL ; )

```

```
{
if (url[strlen(url)-1] == '\n') url[strlen(url)-1] = '\0';

/*
 * Get an array of proxies to use. These should be used
 * in the order returned. Only move on to the next proxy
 * if the first one fails (etc).
 */
print_proxies(px_proxy_factory_get_proxies(pf, url));
}
}
/* Destroy the proxy factory object */
px_proxy_factory_free(pf);
return 0;
}
```


1.77 libtirpc

1.77.1 Available under license

```
/*
 * Copyright (c) Copyright (c) Bull S.A. 2005 All Rights Reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. The name of the author may not be used to endorse or promote products
 * derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR
 * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 * IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
 * THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

/*
 * Copyright (c) 2009, Sun Microsystems, Inc.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 * - Redistributions of source code must retain the above copyright notice,
 * this list of conditions and the following disclaimer.
 * - Redistributions in binary form must reproduce the above copyright notice,
 * this list of conditions and the following disclaimer in the documentation
 * and/or other materials provided with the distribution.
 * - Neither the name of Sun Microsystems, Inc. nor the names of its
 * contributors may be used to endorse or promote products derived
 * from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
```

```

* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*/

/*
* netname utility routines
* convert from unix names to network names and vice-versa
* This module is operating system dependent!
* What we define here will work with any unix system that has adopted
* the sun NIS domain architecture.
*/

#include <sys/param.h>
#include <rpc/rpc.h>
#include "rpc_com.h"
#ifdef YP
#include <rpcsvc/yp_prot.h>
#include <rpcsvc/ypclnt.h>
#endif
#include <ctype.h>
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#ifndef MAXHOSTNAMELEN
#define MAXHOSTNAMELEN 256
#endif
#ifndef NGROUPS
#define NGROUPS 16
#endif

#define TYPE_BIT(type) (sizeof (type) * CHAR_BIT)

#define TYPE_SIGNED(type) (((type) -1) < 0)

/*
** 302 / 1000 is log10(2.0) rounded up.
** Subtract one for the sign bit if the type is signed;
** add one for integer division truncation;
** add one more for a minus sign if the type is signed.
*/
#define INT_STRLEN_MAXIMUM(type) \
    ((TYPE_BIT(type) - TYPE_SIGNED(type)) * 302 / 1000 + 1 + TYPE_SIGNED(type))

static char *OPSYS = "unix";

/*
* Figure out my fully qualified network name
*/
int
getnetname(name)

```

```
char name[MAXNETNAMELEN+1];
{
uid_t uid;

uid = geteuid();
if (uid == 0) {
return (host2netname(name, (char *) NULL, (char *) NULL));
} else {
return (user2netname(name, uid, (char *) NULL));
}
}

/*
 * Convert unix cred to network-name
 */
int
user2netname(netname, uid, domain)
char netname[MAXNETNAMELEN + 1];
const uid_t uid;
const char *domain;
{
char *dfldom;

if (domain == NULL) {
if (__rpc_get_default_domain(&dfldom) != 0) {
return (0);
}
domain = dfldom;
}
if (strlen(domain) + 1 + INT_STRLEN_MAXIMUM(u_long) + 1 + strlen(OPSYS) > MAXNETNAMELEN) {
return (0);
}
(void) sprintf(netname, "%s.%ld@%s", OPSYS, (u_long)uid, domain);
return (1);
}

/*
 * Convert host to network-name
 */
int
host2netname(netname, host, domain)
char netname[MAXNETNAMELEN + 1];
const char *host;
const char *domain;
{
char *dfldom;
char hostname[MAXHOSTNAMELEN+1];

if (domain == NULL) {
if (__rpc_get_default_domain(&dfldom) != 0) {
return (0);
}
}
```

```
domain = dfltdom;
}
if (host == NULL) {
(void) gethostname(hostname, sizeof(hostname));
host = hostname;
}
if (strlen(domain) + 1 + strlen(host) + 1 + strlen(OPSYS) > MAXNETNAMELEN) {
return (0);
}
(void) sprintf(netname, "%s.%s@%s", OPSYS, host, domain);
return (1);
}
```

1.78 libtool

1.78.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid

distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes

a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from

such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a

derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more

than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and

all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.79 libunwind

1.79.1 Available under license

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.80 libusb-compat

1.80.1 Available under license

libusb is covered by the LGPL:

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Alternatively, the files usb.h.in and/or usb.h may be licensed under the BSD license:

Copyright (c) 2000-2003 Johannes Erdfelt <johannes@erdfelt.com>
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.81 libusb1

1.81.1 Available under license

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use,
not price. Our General Public Licenses are designed to make sure that
you have the freedom to distribute copies of free software (and charge
for this service if you wish); that you receive source code or can get
it if you want it; that you can change the software and use pieces of
it in new free programs; and that you are informed that you can do
these things.

To protect your rights, we need to make restrictions that forbid
distributors to deny you these rights or to ask you to surrender these
rights. These restrictions translate to certain responsibilities for
you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them
with the library after making changes to the library and recompiling
it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the
library, and (2) we offer you this license, which gives you legal
permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is

linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a

fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2,

instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6,

whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on

which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not

excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing

and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.82 libxml2

1.82.1 Available under license

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2012 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

```
/*
 * hash.c: chained hash tables
 *
 * Reference: Your favorite introductory book on algorithms
 *
 * Copyright (C) 2000,2012 Bjorn Reese and Daniel Veillard.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED
 * WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF
 * MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE AUTHORS AND
 * CONTRIBUTORS ACCEPT NO RESPONSIBILITY IN ANY CONCEIVABLE MANNER.
 *
 * Author: breese@users.sourceforge.net
 */

#define IN_LIBXML
#include "libxml.h"

#include <string.h>
#ifdef HAVE_STDLIB_H
```

```

#include <stdlib.h>
#endif
#ifdef HAVE_TIME_H
#include <time.h>
#endif

/*
 * Following http://www.ocert.org/advisories/ocert-2011-003.html
 * it seems that having hash randomization might be a good idea
 * when using XML with untrusted data
 */
#ifdef HAVE_RAND && HAVE_SRAND && HAVE_TIME
#define HASH_RANDOMIZATION
#endif

#include <libxml/parser.h>
#include <libxml/hash.h>
#include <libxml/xmlmemory.h>
#include <libxml/xmlerror.h>
#include <libxml/globals.h>

#define MAX_HASH_LEN 8

/* #define DEBUG_GROW */

/*
 * A single entry in the hash table
 */
typedef struct _xmlHashEntry xmlHashEntry;
typedef xmlHashEntry *xmlHashEntryPtr;
struct _xmlHashEntry {
    struct _xmlHashEntry *next;
    xmlChar *name;
    xmlChar *name2;
    xmlChar *name3;
    void *payload;
    int valid;
};

/*
 * The entire hash table
 */
struct _xmlHashTable {
    struct _xmlHashEntry *table;
    int size;
    int nbElems;
    xmlDictPtr dict;
#ifdef HASH_RANDOMIZATION
    int random_seed;
#endif
};

/*
 * xmlHashComputeKey:
 * Calculate the hash key

```

```

*/
static unsigned long
xmlHashComputeKey(xmlHashTablePtr table, const xmlChar *name,
                  const xmlChar *name2, const xmlChar *name3) {
    unsigned long value = 0L;
    char ch;

#ifdef HASH_RANDOMIZATION
    value = table->random_seed;
#endif
    if (name != NULL) {
        value += 30 * (*name);
        while ((ch = *name++) != 0) {
            value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
        }
        value = value ^ ((value << 5) + (value >> 3));
        if (name2 != NULL) {
            while ((ch = *name2++) != 0) {
                value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
            }
            value = value ^ ((value << 5) + (value >> 3));
            if (name3 != NULL) {
                while ((ch = *name3++) != 0) {
                    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
                }
            }
            return (value % table->size);
        }
    }

    static unsigned long
    xmlHashComputeQKey(xmlHashTablePtr table,
                      const xmlChar *prefix, const xmlChar *name,
                      const xmlChar *prefix2, const xmlChar *name2,
                      const xmlChar *prefix3, const xmlChar *name3) {
        unsigned long value = 0L;
        char ch;

#ifdef HASH_RANDOMIZATION
        value = table->random_seed;
#endif
        if (prefix != NULL)
            value += 30 * (*prefix);
        else
            value += 30 * (*name);

        if (prefix != NULL) {
            while ((ch = *prefix++) != 0) {
                value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
            }
            value = value ^ ((value << 5) + (value >> 3) + (unsigned long)'.');
        }
        if (name != NULL) {

```

```

while ((ch = *name++) != 0) {
    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
}
}
value = value ^ ((value << 5) + (value >> 3));
if (prefix2 != NULL) {
while ((ch = *prefix2++) != 0) {
    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
}
value = value ^ ((value << 5) + (value >> 3) + (unsigned long)'.');
}
if (name2 != NULL) {
while ((ch = *name2++) != 0) {
    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
}
}
value = value ^ ((value << 5) + (value >> 3));
if (prefix3 != NULL) {
while ((ch = *prefix3++) != 0) {
    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
}
value = value ^ ((value << 5) + (value >> 3) + (unsigned long)'.');
}
if (name3 != NULL) {
while ((ch = *name3++) != 0) {
    value = value ^ ((value << 5) + (value >> 3) + (unsigned long)ch);
}
}
return (value % table->size);
}

/**
 * xmlHashCreate:
 * @size: the size of the hash table
 *
 * Create a new xmlHashTablePtr.
 *
 * Returns the newly created object, or NULL if an error occurred.
 */
xmlHashTablePtr
xmlHashCreate(int size) {
    xmlHashTablePtr table;

    if (size <= 0)
        size = 256;

    table = xmlMalloc(sizeof(xmlHashTable));
    if (table) {
        table->dict = NULL;
        table->size = size;
    table->nbElems = 0;
        table->table = xmlMalloc(size * sizeof(xmlHashEntry));
        if (table->table) {
            memset(table->table, 0, size * sizeof(xmlHashEntry));
        }
    }
#ifdef HASH_RANDOMIZATION

```

```

        table->random_seed = __xmlRandom();
    #endif
    return(table);
    }
    xmlFree(table);
    }
    return(NULL);
}

/**
 * xmlHashCreateDict:
 * @size: the size of the hash table
 * @dict: a dictionary to use for the hash
 *
 * Create a new xmlHashTablePtr which will use @dict as the internal dictionary
 *
 * Returns the newly created object, or NULL if an error occurred.
 */
xmlHashTablePtr
xmlHashCreateDict(int size, xmlDictPtr dict) {
    xmlHashTablePtr table;

    table = xmlHashCreate(size);
    if (table != NULL) {
        table->dict = dict;
    }
    xmlDictReference(dict);
    return(table);
}

/**
 * xmlHashGrow:
 * @table: the hash table
 * @size: the new size of the hash table
 *
 * resize the hash table
 *
 * Returns 0 in case of success, -1 in case of failure
 */
static int
xmlHashGrow(xmlHashTablePtr table, int size) {
    unsigned long key;
    int oldsize, i;
    xmlHashEntryPtr iter, next;
    struct _xmlHashEntry *oldtable;
    #ifdef DEBUG_GROW
        unsigned long nbElem = 0;
    #endif

    if (table == NULL)
        return(-1);
    if (size < 8)
        return(-1);
    if (size > 8 * 2048)

```

```

return(-1);

    oldsize = table->size;
    oldtable = table->table;
    if (oldtable == NULL)
        return(-1);

    table->table = xmlMalloc(size * sizeof(xmlHashEntry));
    if (table->table == NULL) {
table->table = oldtable;
return(-1);
    }
    memset(table->table, 0, size * sizeof(xmlHashEntry));
    table->size = size;

    /*If the two loops are merged, there would be situations where
    a new entry needs to allocated and data copied into it from
    the main table. So instead, we run through the array twice, first
    copying all the elements in the main array (where we can't get
    conflicts) and then the rest, so we only free (and don't allocate)
    */
    for (i = 0; i < oldsize; i++) {
if (oldtable[i].valid == 0)
    continue;
key = xmlHashComputeKey(table, oldtable[i].name, oldtable[i].name2,
oldtable[i].name3);
memcpy(&(table->table[key]), &(oldtable[i]), sizeof(xmlHashEntry));
table->table[key].next = NULL;
    }

    for (i = 0; i < oldsize; i++) {
iter = oldtable[i].next;
while (iter) {
    next = iter->next;

    /*
    * put back the entry in the new table
    */

    key = xmlHashComputeKey(table, iter->name, iter->name2,
        iter->name3);
    if (table->table[key].valid == 0) {
memcpy(&(table->table[key]), iter, sizeof(xmlHashEntry));
table->table[key].next = NULL;
xmlFree(iter);
    } else {
iter->next = table->table[key].next;
table->table[key].next = iter;
    }

#ifdef DEBUG_GROW
    nbElem++;
#endif

    iter = next;

```

```

    }
    }

    xmlFree(oldtable);

#ifdef DEBUG_GROW
    xmlGenericError(xmlGenericErrorContext,
        "xmlHashGrow : from %d to %d, %d elems\n", oldsize, size, nbElem);
#endif

    return(0);
}

/**
 * xmlHashFree:
 * @table: the hash table
 * @f: the deallocator function for items in the hash
 *
 * Free the hash @table and its contents. The userdata is
 * deallocated with @f if provided.
 */
void
xmlHashFree(xmlHashTablePtr table, xmlHashDeallocator f) {
    int i;
    xmlHashEntryPtr iter;
    xmlHashEntryPtr next;
    int inside_table = 0;
    int nbElems;

    if (table == NULL)
return;
    if (table->table) {
nbElems = table->nbElems;
for(i = 0; (i < table->size) && (nbElems > 0); i++) {
    iter = &(table->table[i]);
    if (iter->valid == 0)
continue;
    inside_table = 1;
    while (iter) {
next = iter->next;
if ((f != NULL) && (iter->payload != NULL))
    f(iter->payload, iter->name);
if (table->dict == NULL) {
    if (iter->name)
xmlFree(iter->name);
    if (iter->name2)
xmlFree(iter->name2);
    if (iter->name3)
xmlFree(iter->name3);
}
iter->payload = NULL;
if (!inside_table)
    xmlFree(iter);
nbElems--;

```

```

        inside_table = 0;
        iter = next;
    }
}
xmlFree(table->table);
}
if (table->dict)
    xmlDictFree(table->dict);
xmlFree(table);
}

/**
 * xmlHashAddEntry:
 * @table: the hash table
 * @name: the name of the userdata
 * @userdata: a pointer to the userdata
 *
 * Add the @userdata to the hash @table. This can later be retrieved
 * by using the @name. Duplicate names generate errors.
 *
 * Returns 0 the addition succeeded and -1 in case of error.
 */
int
xmlHashAddEntry(xmlHashTablePtr table, const xmlChar *name, void *userdata) {
    return(xmlHashAddEntry3(table, name, NULL, NULL, userdata));
}

/**
 * xmlHashAddEntry2:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @userdata: a pointer to the userdata
 *
 * Add the @userdata to the hash @table. This can later be retrieved
 * by using the (@name, @name2) tuple. Duplicate tuples generate errors.
 *
 * Returns 0 the addition succeeded and -1 in case of error.
 */
int
xmlHashAddEntry2(xmlHashTablePtr table, const xmlChar *name,
    const xmlChar *name2, void *userdata) {
    return(xmlHashAddEntry3(table, name, name2, NULL, userdata));
}

/**
 * xmlHashUpdateEntry:
 * @table: the hash table
 * @name: the name of the userdata
 * @userdata: a pointer to the userdata
 * @f: the deallocator function for replaced item (if any)
 *
 * Add the @userdata to the hash @table. This can later be retrieved
 * by using the @name. Existing entry for this @name will be removed
 * and freed with @f if found.

```



```

*
* Returns 0 the addition succeeded and -1 in case of error.
*/
int
xmlHashUpdateEntry(xmlHashTablePtr table, const xmlChar *name,
    void *userdata, xmlHashDeallocator f) {
    return(xmlHashUpdateEntry3(table, name, NULL, NULL, userdata, f));
}

/**
 * xmlHashUpdateEntry2:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @userdata: a pointer to the userdata
 * @f: the deallocator function for replaced item (if any)
 *
 * Add the @userdata to the hash @table. This can later be retrieved
 * by using the (@name, @name2) tuple. Existing entry for this tuple will
 * be removed and freed with @f if found.
 *
 * Returns 0 the addition succeeded and -1 in case of error.
 */
int
xmlHashUpdateEntry2(xmlHashTablePtr table, const xmlChar *name,
    const xmlChar *name2, void *userdata,
    xmlHashDeallocator f) {
    return(xmlHashUpdateEntry3(table, name, name2, NULL, userdata, f));
}

/**
 * xmlHashLookup:
 * @table: the hash table
 * @name: the name of the userdata
 *
 * Find the userdata specified by the @name.
 *
 * Returns the pointer to the userdata
 */
void *
xmlHashLookup(xmlHashTablePtr table, const xmlChar *name) {
    return(xmlHashLookup3(table, name, NULL, NULL));
}

/**
 * xmlHashLookup2:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 *
 * Find the userdata specified by the (@name, @name2) tuple.
 *
 * Returns the pointer to the userdata
 */

```

```

void *
xmlHashLookup2(xmlHashTablePtr table, const xmlChar *name,
    const xmlChar *name2) {
    return(xmlHashLookup3(table, name, name2, NULL));
}

/**
 * xmlHashQLookup:
 * @table: the hash table
 * @prefix: the prefix of the userdata
 * @name: the name of the userdata
 *
 * Find the userdata specified by the QName @prefix:@name/@name.
 *
 * Returns the pointer to the userdata
 */
void *
xmlHashQLookup(xmlHashTablePtr table, const xmlChar *prefix,
    const xmlChar *name) {
    return(xmlHashQLookup3(table, prefix, name, NULL, NULL, NULL, NULL));
}

/**
 * xmlHashQLookup2:
 * @table: the hash table
 * @prefix: the prefix of the userdata
 * @name: the name of the userdata
 * @prefix2: the second prefix of the userdata
 * @name2: a second name of the userdata
 *
 * Find the userdata specified by the QNames tuple
 *
 * Returns the pointer to the userdata
 */
void *
xmlHashQLookup2(xmlHashTablePtr table, const xmlChar *prefix,
    const xmlChar *name, const xmlChar *prefix2,
    const xmlChar *name2) {
    return(xmlHashQLookup3(table, prefix, name, prefix2, name2, NULL, NULL));
}

/**
 * xmlHashAddEntry3:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @name3: a third name of the userdata
 * @userdata: a pointer to the userdata
 *
 * Add the @userdata to the hash @table. This can later be retrieved
 * by using the tuple (@name, @name2, @name3). Duplicate entries generate
 * errors.
 *
 * Returns 0 the addition succeeded and -1 in case of error.
 */

```

```

int
xmlHashAddEntry3(xmlHashTablePtr table, const xmlChar *name,
                 const xmlChar *name2, const xmlChar *name3,
                 void *userdata) {
    unsigned long key, len = 0;
    xmlHashEntryPtr entry;
    xmlHashEntryPtr insert;

    if ((table == NULL) || (name == NULL))
return(-1);

    /*
     * If using a dict internalize if needed
     */
    if (table->dict) {
        if (!xmlDictOwns(table->dict, name)) {
            name = xmlDictLookup(table->dict, name, -1);
            if (name == NULL)
                return(-1);
        }
        if ((name2 != NULL) && (!xmlDictOwns(table->dict, name2))) {
            name2 = xmlDictLookup(table->dict, name2, -1);
            if (name2 == NULL)
                return(-1);
        }
        if ((name3 != NULL) && (!xmlDictOwns(table->dict, name3))) {
            name3 = xmlDictLookup(table->dict, name3, -1);
            if (name3 == NULL)
                return(-1);
        }
    }

    /*
     * Check for duplicate and insertion location.
     */
    key = xmlHashComputeKey(table, name, name2, name3);
    if (table->table[key].valid == 0) {
insert = NULL;
    } else {
        if (table->dict) {
            for (insert = &(table->table[key]); insert->next != NULL;
insert = insert->next) {
if ((insert->name == name) &&
    (insert->name2 == name2) &&
    (insert->name3 == name3))
                return(-1);
len++;
            }
            if ((insert->name == name) &&
(insert->name2 == name2) &&
(insert->name3 == name3))
                return(-1);
        } else {
            for (insert = &(table->table[key]); insert->next != NULL;

```

```

    insert = insert->next) {
    if ((xmlStrEqual(insert->name, name)) &&
        (xmlStrEqual(insert->name2, name2)) &&
        (xmlStrEqual(insert->name3, name3)))
        return(-1);
    len++;
    }
    if ((xmlStrEqual(insert->name, name)) &&
        (xmlStrEqual(insert->name2, name2)) &&
        (xmlStrEqual(insert->name3, name3)))
        return(-1);
    }
    }

    if (insert == NULL) {
    entry = &(table->table[key]);
    } else {
    entry = xmlMalloc(sizeof(xmlHashEntry));
    if (entry == NULL)
        return(-1);
    }

    if (table->dict != NULL) {
        entry->name = (xmlChar *) name;
        entry->name2 = (xmlChar *) name2;
        entry->name3 = (xmlChar *) name3;
    } else {
    entry->name = xmlStrdup(name);
    entry->name2 = xmlStrdup(name2);
    entry->name3 = xmlStrdup(name3);
    }
    entry->payload = userdata;
    entry->next = NULL;
    entry->valid = 1;

    if (insert != NULL)
    insert->next = entry;

    table->nbElems++;

    if (len > MAX_HASH_LEN)
    xmlHashGrow(table, MAX_HASH_LEN * table->size);

    return(0);
}

/**
 * xmlHashUpdateEntry3:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @name3: a third name of the userdata
 * @userdata: a pointer to the userdata
 * @f: the deallocator function for replaced item (if any)

```

```

*
* Add the @userdata to the hash @table. This can later be retrieved
* by using the tuple (@name, @name2, @name3). Existing entry for this tuple
* will be removed and freed with @f if found.
*
* Returns 0 the addition succeeded and -1 in case of error.
*/
int
xmlHashUpdateEntry3(xmlHashTablePtr table, const xmlChar *name,
                    const xmlChar *name2, const xmlChar *name3,
                    void *userdata, xmlHashDeallocator f) {
    unsigned long key;
    xmlHashEntryPtr entry;
    xmlHashEntryPtr insert;

    if ((table == NULL) || name == NULL)
return(-1);

    /*
     * If using a dict internalize if needed
     */
    if (table->dict) {
        if (!xmlDictOwns(table->dict, name)) {
            name = xmlDictLookup(table->dict, name, -1);
            if (name == NULL)
                return(-1);
        }
        if ((name2 != NULL) && (!xmlDictOwns(table->dict, name2))) {
            name2 = xmlDictLookup(table->dict, name2, -1);
            if (name2 == NULL)
                return(-1);
        }
        if ((name3 != NULL) && (!xmlDictOwns(table->dict, name3))) {
            name3 = xmlDictLookup(table->dict, name3, -1);
            if (name3 == NULL)
                return(-1);
        }
    }

    /*
     * Check for duplicate and insertion location.
     */
    key = xmlHashComputeKey(table, name, name2, name3);
    if (table->table[key].valid == 0) {
insert = NULL;
    } else {
        if (table->dict) {
            for (insert = &(table->table[key]); insert->next != NULL;
insert = insert->next) {
if ((insert->name == name) &&
    (insert->name2 == name2) &&
    (insert->name3 == name3)) {
                if (f)
f(insert->payload, insert->name);
            }
        }
    }
}

```

```

        insert->payload = userdata;
        return(0);
    }
    }
    if ((insert->name == name) &&
        (insert->name2 == name2) &&
        (insert->name3 == name3)) {
        if (f)
            f(insert->payload, insert->name);
        insert->payload = userdata;
        return(0);
    }
} else {
    for (insert = &(table->table[key]); insert->next != NULL;
        insert = insert->next) {
        if ((xmlStrEqual(insert->name, name)) &&
            (xmlStrEqual(insert->name2, name2)) &&
            (xmlStrEqual(insert->name3, name3))) {
            if (f)
                f(insert->payload, insert->name);
            insert->payload = userdata;
            return(0);
        }
    }
    if ((xmlStrEqual(insert->name, name)) &&
        (xmlStrEqual(insert->name2, name2)) &&
        (xmlStrEqual(insert->name3, name3))) {
        if (f)
            f(insert->payload, insert->name);
        insert->payload = userdata;
        return(0);
    }
}

if (insert == NULL) {
    entry = &(table->table[key]);
} else {
    entry = xmlMalloc(sizeof(xmlHashEntry));
    if (entry == NULL)
        return(-1);
}

if (table->dict != NULL) {
    entry->name = (xmlChar *) name;
    entry->name2 = (xmlChar *) name2;
    entry->name3 = (xmlChar *) name3;
} else {
    entry->name = xmlStrdup(name);
    entry->name2 = xmlStrdup(name2);
    entry->name3 = xmlStrdup(name3);
}
entry->payload = userdata;
entry->next = NULL;
entry->valid = 1;

```

```

        table->nbElems++;

        if (insert != NULL) {
            insert->next = entry;
        }
        return(0);
    }

/**
 * xmlHashLookup3:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @name3: a third name of the userdata
 *
 * Find the userdata specified by the (@name, @name2, @name3) tuple.
 *
 * Returns the a pointer to the userdata
 */
void *
xmlHashLookup3(xmlHashTablePtr table, const xmlChar *name,
               const xmlChar *name2, const xmlChar *name3) {
    unsigned long key;
    xmlHashEntryPtr entry;

    if (table == NULL)
        return(NULL);
    if (name == NULL)
        return(NULL);
    key = xmlHashComputeKey(table, name, name2, name3);
    if (table->table[key].valid == 0)
        return(NULL);
    if (table->dict) {
        for (entry = &(table->table[key]); entry != NULL; entry = entry->next) {
            if ((entry->name == name) &&
                (entry->name2 == name2) &&
                (entry->name3 == name3))
                return(entry->payload);
        }
        for (entry = &(table->table[key]); entry != NULL; entry = entry->next) {
            if ((xmlStrEqual(entry->name, name)) &&
                (xmlStrEqual(entry->name2, name2)) &&
                (xmlStrEqual(entry->name3, name3)))
                return(entry->payload);
        }
        return(NULL);
    }

/**
 * xmlHashQLookup3:
 * @table: the hash table
 * @prefix: the prefix of the userdata

```

```

* @name: the name of the userdata
* @prefix2: the second prefix of the userdata
* @name2: a second name of the userdata
* @prefix3: the third prefix of the userdata
* @name3: a third name of the userdata
*
* Find the userdata specified by the (@name, @name2, @name3) tuple.
*
* Returns the a pointer to the userdata
*/
void *
xmlHashQLookup3(xmlHashTablePtr table,
                const xmlChar *prefix, const xmlChar *name,
                const xmlChar *prefix2, const xmlChar *name2,
                const xmlChar *prefix3, const xmlChar *name3) {
    unsigned long key;
    xmlHashEntryPtr entry;

    if (table == NULL)
        return(NULL);
    if (name == NULL)
        return(NULL);
    key = xmlHashComputeQKey(table, prefix, name, prefix2,
                            name2, prefix3, name3);
    if (table->table[key].valid == 0)
        return(NULL);
    for (entry = &(table->table[key]); entry != NULL; entry = entry->next) {
        if ((xmlStrQEqual(prefix, name, entry->name)) &&
            (xmlStrQEqual(prefix2, name2, entry->name2)) &&
            (xmlStrQEqual(prefix3, name3, entry->name3)))
            return(entry->payload);
    }
    return(NULL);
}

typedef struct {
    xmlHashScanner hashscanner;
    void *data;
} stubData;

static void
stubHashScannerFull (void *payload, void *data, const xmlChar *name,
                    const xmlChar *name2 ATTRIBUTE_UNUSED,
                    const xmlChar *name3 ATTRIBUTE_UNUSED) {
    stubData *stubdata = (stubData *) data;
    stubdata->hashscanner (payload, stubdata->data, (xmlChar *) name);
}

/**
* xmlHashScan:
* @table: the hash table
* @f: the scanner function for items in the hash
* @data: extra data passed to f
*
* Scan the hash @table and applied @f to each value.

```



```

*/
void
xmlHashScan(xmlHashTablePtr table, xmlHashScanner f, void *data) {
    stubData stubdata;
    stubdata.data = data;
    stubdata.hashscanner = f;
    xmlHashScanFull (table, stubHashScannerFull, &stubdata);
}

/**
 * xmlHashScanFull:
 * @table: the hash table
 * @f: the scanner function for items in the hash
 * @data: extra data passed to f
 *
 * Scan the hash @table and applied @f to each value.
 */
void
xmlHashScanFull(xmlHashTablePtr table, xmlHashScannerFull f, void *data) {
    int i, nb;
    xmlHashEntryPtr iter;
    xmlHashEntryPtr next;

    if (table == NULL)
return;
    if (f == NULL)
return;

    if (table->table) {
for(i = 0; i < table->size; i++) {
    if (table->table[i].valid == 0)
continue;
    iter = &(table->table[i]);
    while (iter) {
next = iter->next;
        nb = table->nbElems;
if ((f != NULL) && (iter->payload != NULL))
    f(iter->payload, data, iter->name,
      iter->name2, iter->name3);
        if (nb != table->nbElems) {
            /* table was modified by the callback, be careful */
            if (iter == &(table->table[i])) {
                if (table->table[i].valid == 0)
                    iter = NULL;
                if (table->table[i].next != next)
                    iter = &(table->table[i]);
            } else
                iter = next;
        } else
            iter = next;
    }
}
}
}
}

```

```

/**
 * xmlHashScan3:
 * @table: the hash table
 * @name: the name of the userdata or NULL
 * @name2: a second name of the userdata or NULL
 * @name3: a third name of the userdata or NULL
 * @f: the scanner function for items in the hash
 * @data: extra data passed to f
 *
 * Scan the hash @table and applied @f to each value matching
 * (@name, @name2, @name3) tuple. If one of the names is null,
 * the comparison is considered to match.
 */
void
xmlHashScan3(xmlHashTablePtr table, const xmlChar *name,
              const xmlChar *name2, const xmlChar *name3,
              xmlHashScanner f, void *data) {
    xmlHashScanFull3 (table, name, name2, name3,
                      (xmlHashScannerFull) f, data);
}

/**
 * xmlHashScanFull3:
 * @table: the hash table
 * @name: the name of the userdata or NULL
 * @name2: a second name of the userdata or NULL
 * @name3: a third name of the userdata or NULL
 * @f: the scanner function for items in the hash
 * @data: extra data passed to f
 *
 * Scan the hash @table and applied @f to each value matching
 * (@name, @name2, @name3) tuple. If one of the names is null,
 * the comparison is considered to match.
 */
void
xmlHashScanFull3(xmlHashTablePtr table, const xmlChar *name,
                  const xmlChar *name2, const xmlChar *name3,
                  xmlHashScannerFull f, void *data) {
    int i;
    xmlHashEntryPtr iter;
    xmlHashEntryPtr next;

    if (table == NULL)
        return;
    if (f == NULL)
        return;

    if (table->table) {
        for(i = 0; i < table->size; i++) {
            if (table->table[i].valid == 0)
                continue;
            iter = &(table->table[i]);
            while (iter) {
                next = iter->next;

```

```

        if (((name == NULL) || (xmlStrEqual(name, iter->name))) &&
            ((name2 == NULL) || (xmlStrEqual(name2, iter->name2))) &&
            ((name3 == NULL) || (xmlStrEqual(name3, iter->name3))) &&
            (iter->payload != NULL)) {
            f(iter->payload, data, iter->name,
              iter->name2, iter->name3);
        }
        iter = next;
    }
}

/**
 * xmlHashCopy:
 * @table: the hash table
 * @f: the copier function for items in the hash
 *
 * Scan the hash @table and applied @f to each value.
 *
 * Returns the new table or NULL in case of error.
 */
xmlHashTablePtr
xmlHashCopy(xmlHashTablePtr table, xmlHashCopier f) {
    int i;
    xmlHashEntryPtr iter;
    xmlHashEntryPtr next;
    xmlHashTablePtr ret;

    if (table == NULL)
        return(NULL);
    if (f == NULL)
        return(NULL);

    ret = xmlHashCreate(table->size);
    if (ret == NULL)
        return(NULL);

    if (table->table) {
        for(i = 0; i < table->size; i++) {
            if (table->table[i].valid == 0)
                continue;
            iter = &(table->table[i]);
            while (iter) {
                next = iter->next;
                xmlHashAddEntry3(ret, iter->name, iter->name2,
                                iter->name3, f(iter->payload, iter->name));
                iter = next;
            }
        }
        ret->nbElems = table->nbElems;
        return(ret);
    }
}

```

```

/**
 * xmlHashSize:
 * @table: the hash table
 *
 * Query the number of elements installed in the hash @table.
 *
 * Returns the number of elements in the hash table or
 * -1 in case of error
 */
int
xmlHashSize(xmlHashTablePtr table) {
    if (table == NULL)
        return(-1);
    return(table->nbElems);
}

/**
 * xmlHashRemoveEntry:
 * @table: the hash table
 * @name: the name of the userdata
 * @f: the deallocator function for removed item (if any)
 *
 * Find the userdata specified by the @name and remove
 * it from the hash @table. Existing userdata for this tuple will be removed
 * and freed with @f.
 *
 * Returns 0 if the removal succeeded and -1 in case of error or not found.
 */
int xmlHashRemoveEntry(xmlHashTablePtr table, const xmlChar *name,
    xmlHashDeallocator f) {
    return(xmlHashRemoveEntry3(table, name, NULL, NULL, f));
}

/**
 * xmlHashRemoveEntry2:
 * @table: the hash table
 * @name: the name of the userdata
 * @name2: a second name of the userdata
 * @f: the deallocator function for removed item (if any)
 *
 * Find the userdata specified by the (@name, @name2) tuple and remove
 * it from the hash @table. Existing userdata for this tuple will be removed
 * and freed with @f.
 *
 * Returns 0 if the removal succeeded and -1 in case of error or not found.
 */
int
xmlHashRemoveEntry2(xmlHashTablePtr table, const xmlChar *name,
    const xmlChar *name2, xmlHashDeallocator f) {
    return(xmlHashRemoveEntry3(table, name, name2, NULL, f));
}

/**
 * xmlHashRemoveEntry3:

```

```

* @table: the hash table
* @name: the name of the userdata
* @name2: a second name of the userdata
* @name3: a third name of the userdata
* @f: the deallocator function for removed item (if any)
*
* Find the userdata specified by the (@name, @name2, @name3) tuple and remove
* it from the hash @table. Existing userdata for this tuple will be removed
* and freed with @f.
*
* Returns 0 if the removal succeeded and -1 in case of error or not found.
*/
int
xmlHashRemoveEntry3(xmlHashTablePtr table, const xmlChar *name,
    const xmlChar *name2, const xmlChar *name3, xmlHashDeallocator f) {
    unsigned long key;
    xmlHashEntryPtr entry;
    xmlHashEntryPtr prev = NULL;

    if (table == NULL || name == NULL)
        return(-1);

    key = xmlHashComputeKey(table, name, name2, name3);
    if (table->table[key].valid == 0) {
        return(-1);
    } else {
        for (entry = &(table->table[key]); entry != NULL; entry = entry->next) {
            if (xmlStrEqual(entry->name, name) &&
                xmlStrEqual(entry->name2, name2) &&
                xmlStrEqual(entry->name3, name3)) {
                if ((f != NULL) && (entry->payload != NULL))
                    f(entry->payload, entry->name);
                entry->payload = NULL;
            }
            if (table->dict == NULL) {
                if(entry->name)
                    xmlFree(entry->name);
                if(entry->name2)
                    xmlFree(entry->name2);
                if(entry->name3)
                    xmlFree(entry->name3);
            }
            if(prev) {
                prev->next = entry->next;
                xmlFree(entry);
            } else {
                if (entry->next == NULL) {
                    entry->valid = 0;
                } else {
                    entry = entry->next;
                    memcpy(&(table->table[key]), entry, sizeof(xmlHashEntry));
                    xmlFree(entry);
                }
            }
            table->nbElems--;
        }
    }
}

```

```

        return(0);
    }
    prev = entry;
}
return(-1);
}
}

#define bottom_hash
#include "elfgcchack.h"
/*
 * list.c: lists handling implementation
 *
 * Copyright (C) 2000 Gary Pennington and Daniel Veillard.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED
 * WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF
 * MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE AUTHORS AND
 * CONTRIBUTORS ACCEPT NO RESPONSIBILITY IN ANY CONCEIVABLE MANNER.
 *
 * Author: Gary.Pennington@uk.sun.com
 */

#define IN_LIBXML
#include "libxml.h"

#include <stdlib.h>
#include <string.h>
#include <libxml/xmlmemory.h>
#include <libxml/list.h>
#include <libxml/globals.h>

/*
 * Type definition are kept internal
 */

struct _xmlLink
{
    struct _xmlLink *next;
    struct _xmlLink *prev;
    void *data;
};

struct _xmlList
{
    xmlLinkPtr sentinel;
    void (*linkDeallocator)(xmlLinkPtr);
    int (*linkCompare)(const void *, const void*);
};

/*****

```

```

*          *
*      Interfaces      *
*          *
*****/

/**
 * xmlLinkDeallocator:
 * @l: a list
 * @lk: a link
 *
 * Unlink and deallocate @lk from list @l
 */
static void
xmlLinkDeallocator(xmlListPtr l, xmlLinkPtr lk)
{
    (lk->prev)->next = lk->next;
    (lk->next)->prev = lk->prev;
    if(l->linkDeallocator)
        l->linkDeallocator(lk);
    xmlFree(lk);
}

/**
 * xmlLinkCompare:
 * @data0: first data
 * @data1: second data
 *
 * Compares two arbitrary data
 *
 * Returns -1, 0 or 1 depending on whether data1 is greater equal or smaller
 *      than data0
 */
static int
xmlLinkCompare(const void *data0, const void *data1)
{
    if (data0 < data1)
        return (-1);
    else if (data0 == data1)
        return (0);
    return (1);
}

/**
 * xmlListLowerSearch:
 * @l: a list
 * @data: a data
 *
 * Search data in the ordered list walking from the beginning
 *
 * Returns the link containing the data or NULL
 */
static xmlLinkPtr
xmlListLowerSearch(xmlListPtr l, void *data)
{

```

```

xmlLinkPtr lk;

if (l == NULL)
    return(NULL);
for(lk = l->sentinel->next; lk != l->sentinel && l->linkCompare(lk->data, data) < 0 ; lk = lk->next);
return lk;
}

/**
 * xmlListHigherSearch:
 * @l: a list
 * @data: a data
 *
 * Search data in the ordered list walking backward from the end
 *
 * Returns the link containing the data or NULL
 */
static xmlLinkPtr
xmlListHigherSearch(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;

    if (l == NULL)
        return(NULL);
    for(lk = l->sentinel->prev; lk != l->sentinel && l->linkCompare(lk->data, data) > 0 ; lk = lk->prev);
    return lk;
}

/**
 * xmlListSearch:
 * @l: a list
 * @data: a data
 *
 * Search data in the list
 *
 * Returns the link containing the data or NULL
 */
static xmlLinkPtr
xmlListLinkSearch(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;
    if (l == NULL)
        return(NULL);
    lk = xmlListLowerSearch(l, data);
    if (lk == l->sentinel)
        return NULL;
    else {
        if (l->linkCompare(lk->data, data) == 0)
            return lk;
        return NULL;
    }
}

/**
 * xmlListLinkReverseSearch:

```



```

* @l: a list
* @data: a data
*
* Search data in the list processing backward
*
* Returns the link containing the data or NULL
*/
static xmlLinkPtr
xmlListLinkReverseSearch(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;
    if (l == NULL)
        return(NULL);
    lk = xmlListHigherSearch(l, data);
    if (lk == l->sentinel)
        return NULL;
    else {
        if (l->linkCompare(lk->data, data) == 0)
            return lk;
        return NULL;
    }
}

/**
* xmlListCreate:
* @dealloc: an optional deallocator function
* @compare: an optional comparison function
*
* Create a new list
*
* Returns the new list or NULL in case of error
*/
xmlListPtr
xmlListCreate(xmlListDeallocator deallocator, xmlListDataCompare compare)
{
    xmlListPtr l;
    if (NULL == (l = (xmlListPtr)xmlMalloc( sizeof(xmlList)))) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for list");
        return (NULL);
    }
    /* Initialize the list to NULL */
    memset(l, 0, sizeof(xmlList));

    /* Add the sentinel */
    if (NULL == (l->sentinel = (xmlLinkPtr)xmlMalloc(sizeof(xmlLink)))) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for sentinel");
    }
    xmlFree(l);
    return (NULL);
}
l->sentinel->next = l->sentinel;
l->sentinel->prev = l->sentinel;
l->sentinel->data = NULL;

```

```

    /* If there is a link deallocator, use it */
    if (deallocator != NULL)
        l->linkDeallocator = deallocator;
    /* If there is a link comparator, use it */
    if (compare != NULL)
        l->linkCompare = compare;
    else /* Use our own */
        l->linkCompare = xmlLinkCompare;
    return l;
}

/**
 * xmlListSearch:
 * @l: a list
 * @data: a search value
 *
 * Search the list for an existing value of @data
 *
 * Returns the value associated to @data or NULL in case of error
 */
void *
xmlListSearch(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;
    if (l == NULL)
        return(NULL);
    lk = xmlListLinkSearch(l, data);
    if (lk)
        return (lk->data);
    return NULL;
}

/**
 * xmlListReverseSearch:
 * @l: a list
 * @data: a search value
 *
 * Search the list in reverse order for an existing value of @data
 *
 * Returns the value associated to @data or NULL in case of error
 */
void *
xmlListReverseSearch(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;
    if (l == NULL)
        return(NULL);
    lk = xmlListLinkReverseSearch(l, data);
    if (lk)
        return (lk->data);
    return NULL;
}

/**

```

```
* xmlListInsert:
* @l: a list
* @data: the data
*
* Insert data in the ordered list at the beginning for this value
*
* Returns 0 in case of success, 1 in case of failure
*/
int
xmlListInsert(xmlListPtr l, void *data)
{
    xmlLinkPtr lkPlace, lkNew;

    if (l == NULL)
        return(1);
    lkPlace = xmlListLowerSearch(l, data);
    /* Add the new link */
    lkNew = (xmlLinkPtr) xmlMalloc(sizeof(xmlLink));
    if (lkNew == NULL) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for new link");
        return (1);
    }
    lkNew->data = data;
    lkPlace = lkPlace->prev;
    lkNew->next = lkPlace->next;
    (lkPlace->next)->prev = lkNew;
    lkPlace->next = lkNew;
    lkNew->prev = lkPlace;
    return 0;
}

/**
* xmlListAppend:
* @l: a list
* @data: the data
*
* Insert data in the ordered list at the end for this value
*
* Returns 0 in case of success, 1 in case of failure
*/
int xmlListAppend(xmlListPtr l, void *data)
{
    xmlLinkPtr lkPlace, lkNew;

    if (l == NULL)
        return(1);
    lkPlace = xmlListHigherSearch(l, data);
    /* Add the new link */
    lkNew = (xmlLinkPtr) xmlMalloc(sizeof(xmlLink));
    if (lkNew == NULL) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for new link");
        return (1);
    }
}
```

```

    }
    lkNew->data = data;
    lkNew->next = lkPlace->next;
    (lkPlace->next)->prev = lkNew;
    lkPlace->next = lkNew;
    lkNew->prev = lkPlace;
    return 0;
}

/**
 * xmlListDelete:
 * @l: a list
 *
 * Deletes the list and its associated data
 */
void xmlListDelete(xmlListPtr l)
{
    if (l == NULL)
        return;

    xmlListClear(l);
    xmlFree(l->sentinel);
    xmlFree(l);
}

/**
 * xmlListRemoveFirst:
 * @l: a list
 * @data: list data
 *
 * Remove the first instance associated to data in the list
 *
 * Returns 1 if a deallocation occurred, or 0 if not found
 */
int
xmlListRemoveFirst(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;

    if (l == NULL)
        return(0);
    /*Find the first instance of this data */
    lk = xmlListLinkSearch(l, data);
    if (lk != NULL) {
        xmlLinkDeallocator(l, lk);
        return 1;
    }
    return 0;
}

/**
 * xmlListRemoveLast:
 * @l: a list
 * @data: list data
 *

```

```
* Remove the last instance associated to data in the list
*
* Returns 1 if a deallocation occurred, or 0 if not found
*/
int
xmlListRemoveLast(xmlListPtr l, void *data)
{
    xmlLinkPtr lk;

    if (l == NULL)
        return(0);
    /*Find the last instance of this data */
    lk = xmlListLinkReverseSearch(l, data);
    if (lk != NULL) {
xmlLinkDeallocator(l, lk);
        return 1;
    }
    return 0;
}

/**
 * xmlListRemoveAll:
 * @l: a list
 * @data: list data
 *
 * Remove the all instance associated to data in the list
 *
 * Returns the number of deallocation, or 0 if not found
 */
int
xmlListRemoveAll(xmlListPtr l, void *data)
{
    int count=0;

    if (l == NULL)
        return(0);

    while(xmlListRemoveFirst(l, data))
        count++;
    return count;
}

/**
 * xmlListClear:
 * @l: a list
 *
 * Remove the all data in the list
 */
void
xmlListClear(xmlListPtr l)
{
    xmlLinkPtr lk;

    if (l == NULL)
```

```

        return;
        lk = l->sentinel->next;
        while(lk != l->sentinel) {
            xmlLinkPtr next = lk->next;

            xmlLinkDeallocator(l, lk);
            lk = next;
        }
    }

/**
 * xmlListEmpty:
 * @l: a list
 *
 * Is the list empty ?
 *
 * Returns 1 if the list is empty, 0 if not empty and -1 in case of error
 */
int
xmlListEmpty(xmlListPtr l)
{
    if(l == NULL)
        return(-1);
    return (l->sentinel->next == l->sentinel);
}

/**
 * xmlListFront:
 * @l: a list
 *
 * Get the first element in the list
 *
 * Returns the first element in the list, or NULL
 */
xmlLinkPtr
xmlListFront(xmlListPtr l)
{
    if(l == NULL)
        return(NULL);
    return (l->sentinel->next);
}

/**
 * xmlListEnd:
 * @l: a list
 *
 * Get the last element in the list
 *
 * Returns the last element in the list, or NULL
 */
xmlLinkPtr
xmlListEnd(xmlListPtr l)
{
    if(l == NULL)
        return(NULL);

```

```
        return (l->sentinel->prev);
    }

    /**
     * xmlListSize:
     * @l: a list
     *
     * Get the number of elements in the list
     *
     * Returns the number of elements in the list or -1 in case of error
     */
    int
    xmlListSize(xmlListPtr l)
    {
        xmlLinkPtr lk;
        int count=0;

        if (l == NULL)
            return(-1);
        /* TODO: keep a counter in xmlList instead */
        for(lk = l->sentinel->next; lk != l->sentinel; lk = lk->next, count++);
        return count;
    }

    /**
     * xmlListPopFront:
     * @l: a list
     *
     * Removes the first element in the list
     */
    void
    xmlListPopFront(xmlListPtr l)
    {
        if(!xmlListEmpty(l))
            xmlLinkDeallocator(l, l->sentinel->next);
    }

    /**
     * xmlListPopBack:
     * @l: a list
     *
     * Removes the last element in the list
     */
    void
    xmlListPopBack(xmlListPtr l)
    {
        if(!xmlListEmpty(l))
            xmlLinkDeallocator(l, l->sentinel->prev);
    }

    /**
     * xmlListPushFront:
     * @l: a list
     * @data: new data
```

```

*
* add the new data at the beginning of the list
*
* Returns 1 if successful, 0 otherwise
*/
int
xmlListPushFront(xmlListPtr l, void *data)
{
    xmlLinkPtr lkPlace, lkNew;

    if (l == NULL)
        return(0);
    lkPlace = l->sentinel;
    /* Add the new link */
    lkNew = (xmlLinkPtr) xmlMalloc(sizeof(xmlLink));
    if (lkNew == NULL) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for new link");
        return (0);
    }
    lkNew->data = data;
    lkNew->next = lkPlace->next;
    (lkPlace->next)->prev = lkNew;
    lkPlace->next = lkNew;
    lkNew->prev = lkPlace;
    return 1;
}

/**
* xmlListPushBack:
* @l: a list
* @data: new data
*
* add the new data at the end of the list
*
* Returns 1 if successful, 0 otherwise
*/
int
xmlListPushBack(xmlListPtr l, void *data)
{
    xmlLinkPtr lkPlace, lkNew;

    if (l == NULL)
        return(0);
    lkPlace = l->sentinel->prev;
    /* Add the new link */
    if (NULL == (lkNew = (xmlLinkPtr) xmlMalloc(sizeof(xmlLink)))) {
        xmlGenericError(xmlGenericErrorContext,
            "Cannot initialize memory for new link");
        return (0);
    }
    lkNew->data = data;
    lkNew->next = lkPlace->next;
    (lkPlace->next)->prev = lkNew;
    lkPlace->next = lkNew;

```



```
    lkNew->prev = lkPlace;
    return 1;
}

/**
 * xmlLinkGetData:
 * @lk: a link
 *
 * See Returns.
 *
 * Returns a pointer to the data referenced from this link
 */
void *
xmlLinkGetData(xmlLinkPtr lk)
{
    if (lk == NULL)
        return(NULL);
    return lk->data;
}

/**
 * xmlListReverse:
 * @l: a list
 *
 * Reverse the order of the elements in the list
 */
void
xmlListReverse(xmlListPtr l)
{
    xmlLinkPtr lk;
    xmlLinkPtr lkPrev;

    if (l == NULL)
        return;
    lkPrev = l->sentinel;
    for (lk = l->sentinel->next; lk != l->sentinel; lk = lk->next) {
        lkPrev->next = lkPrev->prev;
        lkPrev->prev = lk;
        lkPrev = lk;
    }
    /* Fix up the last node */
    lkPrev->next = lkPrev->prev;
    lkPrev->prev = lk;
}

/**
 * xmlListSort:
 * @l: a list
 *
 * Sort all the elements in the list
 */
void
xmlListSort(xmlListPtr l)
{

```

```

xmlListPtr lTemp;

if (l == NULL)
    return;
if(xmlListEmpty(l))
    return;

/* I think that the real answer is to implement quicksort, the
 * alternative is to implement some list copying procedure which
 * would be based on a list copy followed by a clear followed by
 * an insert. This is slow...
 */

if (NULL==(lTemp = xmlListDup(l)))
    return;
xmlListClear(l);
xmlListMerge(l, lTemp);
xmlListDelete(lTemp);
return;
}

/**
 * xmlListWalk:
 * @l: a list
 * @walker: a processing function
 * @user: a user parameter passed to the walker function
 *
 * Walk all the element of the first from first to last and
 * apply the walker function to it
 */
void
xmlListWalk(xmlListPtr l, xmlListWalker walker, const void *user) {
    xmlLinkPtr lk;

    if ((l == NULL) || (walker == NULL))
        return;
    for(lk = l->sentinel->next; lk != l->sentinel; lk = lk->next) {
        if((walker(lk->data, user)) == 0)
            break;
    }
}

/**
 * xmlListReverseWalk:
 * @l: a list
 * @walker: a processing function
 * @user: a user parameter passed to the walker function
 *
 * Walk all the element of the list in reverse order and
 * apply the walker function to it
 */
void
xmlListReverseWalk(xmlListPtr l, xmlListWalker walker, const void *user) {
    xmlLinkPtr lk;

```

```

    if ((l == NULL) || (walker == NULL))
        return;
    for(lk = l->sentinel->prev; lk != l->sentinel; lk = lk->prev) {
        if((walker(lk->data, user)) == 0)
            break;
    }
}

/**
 * xmlListMerge:
 * @l1: the original list
 * @l2: the new list
 *
 * include all the elements of the second list in the first one and
 * clear the second list
 */
void
xmlListMerge(xmlListPtr l1, xmlListPtr l2)
{
    xmlListCopy(l1, l2);
    xmlListClear(l2);
}

/**
 * xmlListDup:
 * @old: the list
 *
 * Duplicate the list
 *
 * Returns a new copy of the list or NULL in case of error
 */
xmlListPtr
xmlListDup(const xmlListPtr old)
{
    xmlListPtr cur;

    if (old == NULL)
        return(NULL);
    /* Hmmm, how to best deal with allocation issues when copying
     * lists. If there is a de-allocator, should responsibility lie with
     * the new list or the old list. Surely not both. I'll arbitrarily
     * set it to be the old list for the time being whilst I work out
     * the answer
     */
    if (NULL == (cur = xmlListCreate(NULL, old->linkCompare)))
        return (NULL);
    if (0 != xmlListCopy(cur, old))
        return NULL;
    return cur;
}

/**
 * xmlListCopy:
 * @cur: the new list

```

```

* @old: the old list
*
* Move all the element from the old list in the new list
*
* Returns 0 in case of success 1 in case of error
*/
int
xmlListCopy(xmlListPtr cur, const xmlListPtr old)
{
    /* Walk the old tree and insert the data into the new one */
    xmlLinkPtr lk;

    if((old == NULL) || (cur == NULL))
        return(1);
    for(lk = old->sentinel->next; lk != old->sentinel; lk = lk->next) {
        if (0 !=xmlListInsert(cur, lk->data)) {
            xmlListDelete(cur);
            return (1);
        }
    }
    return (0);
}
/* xmlListUnique() */
/* xmlListSwap */
#define bottom_list
#include "elfgcchack.h"
/*****
*
* $Id$
*
* Copyright (C) 1998 Bjorn Reese and Daniel Stenberg.
*
* Permission to use, copy, modify, and distribute this software for any
* purpose with or without fee is hereby granted, provided that the above
* copyright notice and this permission notice appear in all copies.
*
* THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF
* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE AUTHORS AND
* CONTRIBUTORS ACCEPT NO RESPONSIBILITY IN ANY CONCEIVABLE MANNER.
*
*****/
*
* A note to trio contributors:
*
* Avoid heap allocation at all costs to ensure that the trio functions
* are async-safe. The exceptions are the printf/fprintf functions, which
* uses fputc, and the asprintf functions and the <alloc> modifier, which
* by design are required to allocate form the heap.
*
*****/
/*
* TODO:
* - Scan is probably too permissive about its modifiers.

```

```

* - C escapes in %#[] ?
* - Multibyte characters (done for format parsing, except scan groups)
* - Complex numbers? (C99 _Complex)
* - Boolean values? (C99 _Bool)
* - C99 NaN(n-char-sequence) missing. The n-char-sequence can be used
*   to print the mantissa, e.g. NaN(0xc000000000000000)
* - Should we support the GNU %a alloc modifier? GNU has an ugly hack
*   for %a, because C99 used %a for other purposes. If specified as
*   %as or %af it is interpreted as the alloc modifier, otherwise as
*   the C99 hex-float. This means that you cannot scan %as as a hex-float
*   immediately followed by an 's'.
* - Scanning of collating symbols.
*/

/*****
* Trio include files
*/
#include "triodef.h"
#include "trio.h"
#include "triop.h"
#include "trionan.h"
#if !defined(TRIO_MINIMAL)
# include "triostr.h"
#endif

/*****
*
* Definitions
*
*****/

#include <math.h>
#include <limits.h>
#include <float.h>

#if (defined(__STDC_ISO_10646__) || defined(MB_LEN_MAX) \
    || defined(USE_MULTIBYTE) || TRIO_WIDECHAR) \
    && !defined(_WIN32_WCE)
# define TRIO_COMPILER_SUPPORTS_MULTIBYTE
# if !defined(MB_LEN_MAX)
# define MB_LEN_MAX 6
# endif
#endif

#if (defined(TRIO_COMPILER_MSVC) && (_MSC_VER >= 1100)) || defined(TRIO_COMPILER_BCB)
# define TRIO_COMPILER_SUPPORTS_MSVC_INT
#endif

#if defined(_WIN32_WCE)
#include <wincompat.h>
#endif

/*****
* Generic definitions

```

```

*/

#if !(defined(DEBUG) || defined(NDEBUG))
# define NDEBUG
#endif

#include <assert.h>
#include <ctype.h>
#if !defined(TRIO_COMPILER_SUPPORTS_C99)
# define isblank(x) (((x)==32) || ((x)==9))
#endif
#if defined(TRIO_COMPILER_ANCIENT)
# include <varargs.h>
#else
# include <stdarg.h>
#endif
#include <stddef.h>

#if defined( HAVE_ERRNO_H ) || defined( __VMS )
#include <errno.h>
#endif

#ifndef NULL
# define NULL 0
#endif
#define NIL ((char)0)
#ifndef FALSE
# define FALSE (1 == 0)
# define TRUE (! FALSE)
#endif
#define BOOLEAN_T int

/* mincore() can be used for debugging purposes */
#define VALID(x) (NULL != (x))

#if TRIO_ERRORS
/*
 * Encode the error code and the position. This is decoded
 * with TRIO_ERROR_CODE and TRIO_ERROR_POSITION.
 */
# define TRIO_ERROR_RETURN(x,y) (- ((x) + ((y) << 8)))
#else
# define TRIO_ERROR_RETURN(x,y) (-1)
#endif

#ifndef VA_LIST_IS_ARRAY
#define TRIO_VA_LIST_PTRva_list *
#define TRIO_VA_LIST_ADDR(l)(&(l))
#define TRIO_VA_LIST_DEREF(l)(*(l))
#else
#define TRIO_VA_LIST_PTRva_list
#define TRIO_VA_LIST_ADDR(l)(l)
#define TRIO_VA_LIST_DEREF(l)(l)
#endif

```

```
typedef unsigned long trio_flags_t;
```

```

/*****
 * Platform specific definitions
 */
#if defined(TRIO_PLATFORM_UNIX) || defined(TRIO_PLATFORM_OS400)
# include <unistd.h>
# include <signal.h>
# include <locale.h>
# define USE_LOCALE
#endif /* TRIO_PLATFORM_UNIX */
#if defined(TRIO_PLATFORM_VMS)
# include <unistd.h>
#endif
#if defined(TRIO_PLATFORM_WIN32)
# if defined(_WIN32_WCE)
# include <winccompat.h>
# else
# include <io.h>
# define read _read
# define write _write
# endif
#endif /* TRIO_PLATFORM_WIN32 */

#if TRIO_WIDECHAR
# if defined(TRIO_COMPILER_SUPPORTS_ISO94)
# include <wchar.h>
# include <wctype.h>
typedef wchar_t trio_wchar_t;
typedef wint_t trio_wint_t;
# else
typedef char trio_wchar_t;
typedef int trio_wint_t;
# define WCONST(x) L ## x
# define WEOF EOF
# define iswalnum(x) isalnum(x)
# define iswalphalpha(x) isalpha(x)
# define iswblank(x) isblank(x)
# define iswcntrl(x) iscntrl(x)
# define iswdigit(x) isdigit(x)
# define iswgraph(x) isgraph(x)
# define iswlower(x) islower(x)
# define iswprint(x) isprint(x)
# define iswpunct(x) ispunct(x)
# define iswspace(x) isspace(x)
# define iswupper(x) isupper(x)
# define iswxdigit(x) isxdigit(x)
# endif
#endif

/*****
 * Compiler dependent definitions

```

```

*/

/* Support for long long */
#ifndef __cplusplus
# if !defined(USE_LONGLONG)
#  if defined(TRIO_COMPILER_GCC) && !defined(__STRICT_ANSI__)
#   define USE_LONGLONG
#  elif defined(TRIO_COMPILER_SUNPRO)
#   define USE_LONGLONG
#  elif defined(_LONG_LONG) || defined(_LONGLONG)
#   define USE_LONGLONG
#  endif
# endif
#endif

/* The extra long numbers */
#if defined(USE_LONGLONG)
typedef signed long long int trio_longlong_t;
typedef unsigned long long int trio_ulonglong_t;
#elif defined(TRIO_COMPILER_SUPPORTS_MSVC_INT)
typedef signed __int64 trio_longlong_t;
typedef unsigned __int64 trio_ulonglong_t;
#else
typedef TRIO_SIGNED long int trio_longlong_t;
typedef unsigned long int trio_ulonglong_t;
#endif

/* Maximal and fixed integer types */
#if defined(TRIO_COMPILER_SUPPORTS_C99) && !defined(__VMS)
# include <stdint.h>
typedef intmax_t trio_intmax_t;
typedef uintmax_t trio_uintmax_t;
typedef int8_t trio_int8_t;
typedef int16_t trio_int16_t;
typedef int32_t trio_int32_t;
typedef int64_t trio_int64_t;
#elif defined(TRIO_COMPILER_SUPPORTS_UNIX98) || defined(__VMS)
# include <inttypes.h>
#ifdef __VMS
typedef long long int intmax_t;
typedef unsigned long long int uintmax_t;
#endif
typedef intmax_t trio_intmax_t;
typedef uintmax_t trio_uintmax_t;
typedef int8_t trio_int8_t;
typedef int16_t trio_int16_t;
typedef int32_t trio_int32_t;
typedef int64_t trio_int64_t;
#elif defined(TRIO_COMPILER_SUPPORTS_MSVC_INT)
typedef trio_longlong_t trio_intmax_t;
typedef trio_ulonglong_t trio_uintmax_t;
typedef __int8 trio_int8_t;
typedef __int16 trio_int16_t;
typedef __int32 trio_int32_t;
typedef __int64 trio_int64_t;

```



```

#else
typedef trio_longlong_t trio_intmax_t;
typedef trio_ulonglong_t trio_uintmax_t;
# if defined(TRIO_INT8_T)
typedef TRIO_INT8_T trio_int8_t;
# else
typedef TRIO_SIGNED char trio_int8_t;
# endif
# if defined(TRIO_INT16_T)
typedef TRIO_INT16_T trio_int16_t;
# else
typedef TRIO_SIGNED short trio_int16_t;
# endif
# if defined(TRIO_INT32_T)
typedef TRIO_INT32_T trio_int32_t;
# else
typedef TRIO_SIGNED int trio_int32_t;
# endif
# if defined(TRIO_INT64_T)
typedef TRIO_INT64_T trio_int64_t;
# else
typedef trio_longlong_t trio_int64_t;
# endif
#endif

#if (!(defined(TRIO_COMPILER_SUPPORTS_C99) \
|| defined(TRIO_COMPILER_SUPPORTS_UNIX01))) \
&& !defined(_WIN32_WCE)
# define floorl(x) floor((double)(x))
# define fmodl(x,y) fmod((double)(x),(double)(y))
# define powl(x,y) pow((double)(x),(double)(y))
#endif

#define TRIO_FABS(x) (((x) < 0.0) ? -(x) : (x))

/*****
 * Internal Definitions
 */

#ifndef DECIMAL_DIG
# define DECIMAL_DIG DBL_DIG
#endif

/* Long double sizes */
#ifndef LDBL_DIG
# define MAX_MANTISSA_DIGITS LDBL_DIG
# define MAX_EXPONENT_DIGITS 4
# define MAX_DOUBLE_DIGITS LDBL_MAX_10_EXP
#else
# define MAX_MANTISSA_DIGITS DECIMAL_DIG
# define MAX_EXPONENT_DIGITS 3
# define MAX_DOUBLE_DIGITS DBL_MAX_10_EXP
#endif

```

```

#if defined(TRIO_COMPILER_ANCIENT) || !defined(LDBL_DIG)
# undef LDBL_DIG
# undef LDBL_MANT_DIG
# undef LDBL_EPSILON
# define LDBL_DIG DBL_DIG
# define LDBL_MANT_DIG DBL_MANT_DIG
# define LDBL_EPSILON DBL_EPSILON
#endif

/* The maximal number of digits is for base 2 */
#define MAX_CHARS_IN(x) (sizeof(x) * CHAR_BIT)
/* The width of a pointer. The number of bits in a hex digit is 4 */
#define POINTER_WIDTH ((sizeof("0x") - 1) + sizeof(trio_pointer_t) * CHAR_BIT / 4)

/* Infinite and Not-A-Number for floating-point */
#define INFINITE_LOWER "inf"
#define INFINITE_UPPER "INF"
#define LONG_INFINITE_LOWER "infinite"
#define LONG_INFINITE_UPPER "INFINITE"
#define NAN_LOWER "nan"
#define NAN_UPPER "NAN"

#if !defined(HAVE_ISASCII) && !defined(isascii)
#ifdef __VMS
# define isascii(x) ((unsigned int)(x) < 128)
#endif
#endif

/* Various constants */
enum {
    TYPE_PRINT = 1,
    TYPE_SCAN = 2,

    /* Flags. FLAGS_LAST must be less than ULONG_MAX */
    FLAGS_NEW = 0,
    FLAGS_STICKY = 1,
    FLAGS_SPACE = 2 * FLAGS_STICKY,
    FLAGS_SHOWSIGN = 2 * FLAGS_SPACE,
    FLAGS_LEFTADJUST = 2 * FLAGS_SHOWSIGN,
    FLAGS_ALTERNATIVE = 2 * FLAGS_LEFTADJUST,
    FLAGS_SHORT = 2 * FLAGS_ALTERNATIVE,
    FLAGS_SHORTEST = 2 * FLAGS_SHORT,
    FLAGS_LONG = 2 * FLAGS_SHORTEST,
    FLAGS_QUAD = 2 * FLAGS_LONG,
    FLAGS_LONGDOUBLE = 2 * FLAGS_QUAD,
    FLAGS_SIZE_T = 2 * FLAGS_LONGDOUBLE,
    FLAGS_PTRDIFF_T = 2 * FLAGS_SIZE_T,
    FLAGS_INTMAX_T = 2 * FLAGS_PTRDIFF_T,
    FLAGS_NILPADDING = 2 * FLAGS_INTMAX_T,
    FLAGS_UNSIGNED = 2 * FLAGS_NILPADDING,
    FLAGS_UPPER = 2 * FLAGS_UNSIGNED,
    FLAGS_WIDTH = 2 * FLAGS_UPPER,
    FLAGS_WIDTH_PARAMETER = 2 * FLAGS_WIDTH,
    FLAGS_PRECISION = 2 * FLAGS_WIDTH_PARAMETER,
    FLAGS_PRECISION_PARAMETER = 2 * FLAGS_PRECISION,

```

```

    FLAGS_BASE          = 2 * FLAGS_PRECISION_PARAMETER,
    FLAGS_BASE_PARAMETER = 2 * FLAGS_BASE,
    FLAGS_FLOAT_E        = 2 * FLAGS_BASE_PARAMETER,
    FLAGS_FLOAT_G        = 2 * FLAGS_FLOAT_E,
    FLAGS_QUOTE          = 2 * FLAGS_FLOAT_G,
    FLAGS_WIDECHAR        = 2 * FLAGS_QUOTE,
    FLAGS_ALLOC          = 2 * FLAGS_WIDECHAR,
    FLAGS_IGNORE         = 2 * FLAGS_ALLOC,
    FLAGS_IGNORE_PARAMETER = 2 * FLAGS_IGNORE,
    FLAGS_VARSIZE_PARAMETER = 2 * FLAGS_IGNORE_PARAMETER,
    FLAGS_FIXED_SIZE     = 2 * FLAGS_VARSIZE_PARAMETER,
    FLAGS_LAST           = FLAGS_FIXED_SIZE,
    /* Reused flags */
    FLAGS_EXCLUDE        = FLAGS_SHORT,
    FLAGS_USER_DEFINED   = FLAGS_IGNORE,
    FLAGS_ROUNDING       = FLAGS_INTMAX_T,
    /* Compounded flags */
    FLAGS_ALL_VARSIZES   = FLAGS_LONG | FLAGS_QUAD | FLAGS_INTMAX_T | FLAGS_PTRDIFF-
    F_T | FLAGS_SIZE_T,
    FLAGS_ALL_SIZES      = FLAGS_ALL_VARSIZES | FLAGS_SHORTEST | FLAGS_SHORT,

    NO_POSITION = -1,
    NO_WIDTH    = 0,
    NO_PRECISION = -1,
    NO_SIZE     = -1,

    /* Do not change these */
    NO_BASE    = -1,
    MIN_BASE   = 2,
    MAX_BASE   = 36,
    BASE_BINARY = 2,
    BASE_OCTAL  = 8,
    BASE_DECIMAL = 10,
    BASE_HEX    = 16,

    /* Maximal number of allowed parameters */
    MAX_PARAMETERS = 64,
    /* Maximal number of characters in class */
    MAX_CHARACTER_CLASS = UCHAR_MAX + 1,

    /* Maximal string lengths for user-defined specifiers */
    MAX_USER_NAME = 64,
    MAX_USER_DATA = 256,

    /* Maximal length of locale separator strings */
    MAX_LOCALE_SEPARATOR_LENGTH = MB_LEN_MAX,
    /* Maximal number of integers in grouping */
    MAX_LOCALE_GROUPS = 64,

    /* Initial size of asprintf buffer */
    DYNAMIC_START_SIZE = 32
};

#define NO_GROUPING ((int)CHAR_MAX)

```

```

/* Fundamental formatting parameter types */
#define FORMAT_UNKNOWN 0
#define FORMAT_INT 1
#define FORMAT_DOUBLE 2
#define FORMAT_CHAR 3
#define FORMAT_STRING 4
#define FORMAT_POINTER 5
#define FORMAT_COUNT 6
#define FORMAT_PARAMETER 7
#define FORMAT_GROUP 8
#if TRIO_GNU
# define FORMAT_ERRNO 9
#endif
#if TRIO_EXTENSION
# define FORMAT_USER_DEFINED 10
#endif

/* Character constants */
#define CHAR_IDENTIFIER '%'
#define CHAR_BACKSLASH '\\'
#define CHAR_QUOTE '"'
#define CHAR_ADJUST ' '

/* Character class expressions */
#define CLASS_ALNUM "[:alnum:]"
#define CLASS_ALPHA "[:alpha:]"
#define CLASS_BLANK "[:blank:]"
#define CLASS_CNTRL "[:cntrl:]"
#define CLASS_DIGIT "[:digit:]"
#define CLASS_GRAPH "[:graph:]"
#define CLASS_LOWER "[:lower:]"
#define CLASS_PRINT "[:print:]"
#define CLASS_PUNCT "[:punct:]"
#define CLASS_SPACE "[:space:]"
#define CLASS_UPPER "[:upper:]"
#define CLASS_XDIGIT "[:xdigit:]"

/*
* SPECIFIERS:
*
*
* a Hex-float
* A Hex-float
* c Character
* C Widechar character (wint_t)
* d Decimal
* e Float
* E Float
* F Float
* F Float
* g Float
* G Float
* i Integer
* m Error message

```

```

* n Count
* o Octal
* p Pointer
* s String
* S Widechar string (wchar_t *)
* u Unsigned
* x Hex
* X Hex
* [] Group
* <> User-defined
*
* Reserved:
*
* D Binary Coded Decimal %D(length,precision) (OS/390)
*/
#define SPECIFIER_CHAR 'c'
#define SPECIFIER_STRING 's'
#define SPECIFIER_DECIMAL 'd'
#define SPECIFIER_INTEGER 'i'
#define SPECIFIER_UNSIGNED 'u'
#define SPECIFIER_OCTAL 'o'
#define SPECIFIER_HEX 'x'
#define SPECIFIER_HEX_UPPER 'X'
#define SPECIFIER_FLOAT_E 'e'
#define SPECIFIER_FLOAT_E_UPPER 'E'
#define SPECIFIER_FLOAT_F 'f'
#define SPECIFIER_FLOAT_F_UPPER 'F'
#define SPECIFIER_FLOAT_G 'g'
#define SPECIFIER_FLOAT_G_UPPER 'G'
#define SPECIFIER_POINTER 'p'
#define SPECIFIER_GROUP '['
#define SPECIFIER_UNGROUP ']'
#define SPECIFIER_COUNT 'n'
#if TRIO_UNIX98
# define SPECIFIER_CHAR_UPPER 'C'
# define SPECIFIER_STRING_UPPER 'S'
#endif
#if TRIO_C99
# define SPECIFIER_HEXFLOAT 'a'
# define SPECIFIER_HEXFLOAT_UPPER 'A'
#endif
#if TRIO_GNU
# define SPECIFIER_ERRNO 'm'
#endif
#if TRIO_EXTENSION
# define SPECIFIER_BINARY 'b'
# define SPECIFIER_BINARY_UPPER 'B'
# define SPECIFIER_USER_DEFINED_BEGIN '<'
# define SPECIFIER_USER_DEFINED_END '>'
# define SPECIFIER_USER_DEFINED_SEPARATOR ':'
#endif

/*
* QUALIFIERS:

```

```

*
*
* Numbers = d,i,o,u,x,X
* Float = a,A,e,E,f,F,g,G
* String = s
* Char = c
*
*
* 9$ Position
*   Use the 9th parameter. 9 can be any number between 1 and
*   the maximal argument
*
* 9 Width
*   Set width to 9. 9 can be any number, but must not be postfix
*   by '$'
*
* h Short
*   Numbers:
*   (unsigned) short int
*
* hh Short short
*   Numbers:
*   (unsigned) char
*
* l Long
*   Numbers:
*   (unsigned) long int
*   String:
*   as the S specifier
*   Char:
*   as the C specifier
*
* ll Long Long
*   Numbers:
*   (unsigned) long long int
*
* L Long Double
*   Float
*   long double
*
* # Alternative
*   Float:
*   Decimal-point is always present
*   String:
*   non-printable characters are handled as \number
*
* Spacing
*
* + Sign
*
* - Alignment
*
* . Precision
*
* * Parameter

```

```

* print: use parameter
* scan: no parameter (ignore)
*
* q Quad
*
* Z size_t
*
* w Widechar
*
* ' Thousands/quote
* Numbers:
* Integer part grouped in thousands
* Binary numbers:
* Number grouped in nibbles (4 bits)
* String:
* Quoted string
*
* j intmax_t
* t ptrdiff_t
* z size_t
*
* ! Sticky
* @ Parameter (for both print and scan)
*
* I n-bit Integer
* Numbers:
* The following options exists
* I8 = 8-bit integer
* I16 = 16-bit integer
* I32 = 32-bit integer
* I64 = 64-bit integer
*/
#define QUALIFIER_POSITION '$'
#define QUALIFIER_SHORT 'h'
#define QUALIFIER_LONG 'l'
#define QUALIFIER_LONG_UPPER 'L'
#define QUALIFIER_ALTERNATIVE '#'
#define QUALIFIER_SPACE ' '
#define QUALIFIER_PLUS '+'
#define QUALIFIER_MINUS '-'
#define QUALIFIER_DOT '.'
#define QUALIFIER_STAR '*'
#define QUALIFIER_CIRCUMFLEX '^' /* For scanlists */
#if TRIO_C99
# define QUALIFIER_SIZE_T 'z'
# define QUALIFIER_PTRDIFF_T 't'
# define QUALIFIER_INTMAX_T 'j'
#endif
#if TRIO_BSD || TRIO_GNU
# define QUALIFIER_QUAD 'q'
#endif
#if TRIO_GNU
# define QUALIFIER_SIZE_T_UPPER 'Z'
#endif

```

```

#if TRIO_MISC
# define QUALIFIER_WIDECHAR 'w'
#endif
#if TRIO_MICROSOFT
# define QUALIFIER_FIXED_SIZE 'T'
#endif
#if TRIO_EXTENSION
# define QUALIFIER_QUOTE "\""
# define QUALIFIER_STICKY '!'
# define QUALIFIER_VARSIZE '&' /* This should remain undocumented */
# define QUALIFIER_PARAM '@' /* Experimental */
# define QUALIFIER_COLON ':' /* For scanlists */
# define QUALIFIER_EQUAL '=' /* For scanlists */
# define QUALIFIER_ROUNDING_UPPER 'R'
#endif

/*****
*
* Internal Structures
*
*****/

/* Parameters */
typedef struct {
    /* An indication of which entry in the data union is used */
    int type;
    /* The flags */
    trio_flags_t flags;
    /* The width qualifier */
    int width;
    /* The precision qualifier */
    int precision;
    /* The base qualifier */
    int base;
    /* The size for the variable size qualifier */
    int varsize;
    /* The marker of the end of the specifier */
    int indexAfterSpecifier;
    /* The data from the argument list */
    union {
        char *string;
#if TRIO_WIDECHAR
        trio_wchar_t *wstring;
#endif
        trio_pointer_t pointer;
        union {
            trio_intmax_t as_signed;
            trio_uintmax_t as_unsigned;
        } number;
        double doubleNumber;
        double *doublePointer;
        trio_long_double_t longdoubleNumber;
        trio_long_double_t *longdoublePointer;
        int errorNumber;
    };
};

```



```
    } data;
    /* For the user-defined specifier */
    char user_name[MAX_USER_NAME];
    char user_data[MAX_USER_DATA];
} trio_parameter_t;

/* Container for customized functions */
typedef struct {
    union {
        trio_outstream_t out;
        trio_instream_t in;
    } stream;
    trio_pointer_t closure;
} trio_custom_t;

/* General trio "class" */
typedef struct _trio_class_t {
    /*
     * The function to write characters to a stream.
     */
    void (*OutStream) TRIO_PROTO((struct _trio_class_t *, int));
    /*
     * The function to read characters from a stream.
     */
    void (*InStream) TRIO_PROTO((struct _trio_class_t *, int *));
    /*
     * The current location in the stream.
     */
    trio_pointer_t location;
    /*
     * The character currently being processed.
     */
    int current;
    /*
     * The number of characters that would have been written/read
     * if there had been sufficient space.
     */
    int processed;
    /*
     * The number of characters that are actually written/read.
     * Processed and committed will only differ for the *nprintf
     * and *nscanf functions.
     */
    int committed;
    /*
     * The upper limit of characters that may be written/read.
     */
    int max;
    /*
     * The last output error that was detected.
     */
    int error;
} trio_class_t;
```

```

/* References (for user-defined callbacks) */
typedef struct _trio_reference_t {
    trio_class_t *data;
    trio_parameter_t *parameter;
} trio_reference_t;

/* Registered entries (for user-defined callbacks) */
typedef struct _trio_userdef_t {
    struct _trio_userdef_t *next;
    trio_callback_t callback;
    char *name;
} trio_userdef_t;

/*****
 *
 * Internal Variables
 *
 *****/

static TRIO_CONST char rcsid[] = "@(#) $Id$";

/*
 * Need this to workaround a parser bug in HP C/iX compiler that fails
 * to resolves macro definitions that includes type 'long double',
 * e.g: va_arg(arg_ptr, long double)
 */
#ifdef TRIO_PLATFORM_MPEIX
static TRIO_CONST trio_long_double_t ___dummy_long_double = 0;
#endif

static TRIO_CONST char internalNullString[] = "(nil)";

#ifdef USE_LOCALE
static struct lconv *internalLocaleValues = NULL;
#endif

/*
 * UNIX98 says "in a locale where the radix character is not defined,
 * the radix character defaults to a period (.)"
 */
static int internalDecimalPointLength = 1;
static int internalThousandSeparatorLength = 1;
static char internalDecimalPoint = '!';
static char internalDecimalPointString[MAX_LOCALE_SEPARATOR_LENGTH + 1] = ".";
static char internalThousandSeparator[MAX_LOCALE_SEPARATOR_LENGTH + 1] = ",";
static char internalGrouping[MAX_LOCALE_GROUPS] = { (char)NO_GROUPING };

static TRIO_CONST char internalDigitsLower[] = "0123456789abcdefghijklmnopqrstuvwxyz";
static TRIO_CONST char internalDigitsUpper[] = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";
static BOOLEAN_T internalDigitsUnconverted = TRUE;
static int internalDigitArray[128];
#ifdef TRIO_EXTENSION
static BOOLEAN_T internalCollationUnconverted = TRUE;
static char internalCollationArray[MAX_CHARACTER_CLASS][MAX_CHARACTER_CLASS];
#endif

```

```

#if TRIO_EXTENSION
static TRIO_VOLATILE trio_callback_t internalEnterCriticalRegion = NULL;
static TRIO_VOLATILE trio_callback_t internalLeaveCriticalRegion = NULL;
static trio_userdef_t *internalUserDef = NULL;
#endif

/*****
 *
 * Internal Functions
 *
 *****/

#if defined(TRIO_MINIMAL)
# define TRIO_STRING_PUBLIC static
# include "triostr.c"
#endif /* defined(TRIO_MINIMAL) */

/*****
 * TrioIsQualifier
 *
 * Description:
 * Remember to add all new qualifiers to this function.
 * QUALIFIER_POSITION must not be added.
 */
TRIO_PRIVATE BOOLEAN_T
TrioIsQualifier
TRIO_ARGS1((character),
    TRIO_CONST char character)
{
    /* QUALIFIER_POSITION is not included */
    switch (character)
    {
        case '0': case '1': case '2': case '3': case '4':
        case '5': case '6': case '7': case '8': case '9':
        case QUALIFIER_PLUS:
        case QUALIFIER_MINUS:
        case QUALIFIER_SPACE:
        case QUALIFIER_DOT:
        case QUALIFIER_STAR:
        case QUALIFIER_ALTERNATIVE:
        case QUALIFIER_SHORT:
        case QUALIFIER_LONG:
        case QUALIFIER_LONG_UPPER:
        case QUALIFIER_CIRCUMFLEX:
        #if defined(QUALIFIER_SIZE_T)
        case QUALIFIER_SIZE_T:
        #endif
        #if defined(QUALIFIER_PTRDIFF_T)
        case QUALIFIER_PTRDIFF_T:
        #endif
        #if defined(QUALIFIER_INTMAX_T)
        case QUALIFIER_INTMAX_T:

```

```

#endif
#if defined(QUALIFIER_QUAD)
    case QUALIFIER_QUAD:
#endif
#if defined(QUALIFIER_SIZE_T_UPPER)
    case QUALIFIER_SIZE_T_UPPER:
#endif
#if defined(QUALIFIER_WIDECHAR)
    case QUALIFIER_WIDECHAR:
#endif
#if defined(QUALIFIER_QUOTE)
    case QUALIFIER_QUOTE:
#endif
#if defined(QUALIFIER_STICKY)
    case QUALIFIER_STICKY:
#endif
#if defined(QUALIFIER_VARSIZE)
    case QUALIFIER_VARSIZE:
#endif
#if defined(QUALIFIER_PARAM)
    case QUALIFIER_PARAM:
#endif
#if defined(QUALIFIER_FIXED_SIZE)
    case QUALIFIER_FIXED_SIZE:
#endif
#if defined(QUALIFIER_ROUNDING_UPPER)
    case QUALIFIER_ROUNDING_UPPER:
#endif
    return TRUE;
default:
    return FALSE;
}
}

/*****
* TrioSetLocale
*/
#if defined(USE_LOCALE)
TRIO_PRIVATE void
TrioSetLocale(TRIO_NOARGS)
{
    internalLocaleValues = (struct lconv *)localeconv();
    if (internalLocaleValues)
    {
        if ((internalLocaleValues->decimal_point) &&
            (internalLocaleValues->decimal_point[0] != NIL))
        {
            internalDecimalPointLength = trio_length(internalLocaleValues->decimal_point);
            if (internalDecimalPointLength == 1)
            {
                internalDecimalPoint = internalLocaleValues->decimal_point[0];
            }
        }
        else
        {
            internalDecimalPoint = NIL;

```

```

        trio_copy_max(internalDecimalPointString,
        sizeof(internalDecimalPointString),
        internalLocaleValues->decimal_point);
    }
}
    if ((internalLocaleValues->thousands_sep) &&
        (internalLocaleValues->thousands_sep[0] != NIL))
    {
        trio_copy_max(internalThousandSeparator,
        sizeof(internalThousandSeparator),
        internalLocaleValues->thousands_sep);
        internalThousandSeparatorLength = trio_length(internalThousandSeparator);
    }
    if ((internalLocaleValues->grouping) &&
        (internalLocaleValues->grouping[0] != NIL))
    {
        trio_copy_max(internalGrouping,
        sizeof(internalGrouping),
        internalLocaleValues->grouping);
    }
}
}
#endif /* defined(USE_LOCALE) */

```

```

TRIO_PRIVATE int
TrioCalcThousandSeparatorLength
TRIO_ARGS1((digits),
    int digits)
{
#ifdef TRIO_EXTENSION
    int count = 0;
    int step = NO_GROUPING;
    char *groupingPointer = internalGrouping;

    while (digits > 0)
    {
        if (*groupingPointer == CHAR_MAX)
        {
            /* Disable grouping */
            break; /* while */
        }
        else if (*groupingPointer == 0)
        {
            /* Repeat last group */
            if (step == NO_GROUPING)
            {
                /* Error in locale */
                break; /* while */
            }
        }
        else
        {
            step = *groupingPointer++;
        }
    }
}

```

```

        if (digits > step)
count += internalThousandSeparatorLength;
        digits -= step;
    }
    return count;
#else
    return 0;
#endif
}

TRIO_PRIVATE BOOLEAN_T
TrioFollowedBySeparator
TRIO_ARGS1((position),
    int position)
{
#ifdef TRIO_EXTENSION
    int step = 0;
    char *groupingPointer = internalGrouping;

    position--;
    if (position == 0)
        return FALSE;
    while (position > 0)
    {
        if (*groupingPointer == CHAR_MAX)
        {
            /* Disable grouping */
            break; /* while */
        }
        else if (*groupingPointer != 0)
        {
            step = *groupingPointer++;
        }
        if (step == 0)
            break;
        position -= step;
    }
    return (position == 0);
#else
    return FALSE;
#endif
}

/*****
* TrioGetPosition
*
* Get the %n$ position.
*/
TRIO_PRIVATE int
TrioGetPosition
TRIO_ARGS2((format, indexPointer),
    TRIO_CONST char *format,
    int *indexPointer)
{
#ifdef TRIO_UNIX98

```

```

char *tmpformat;
int number = 0;
int index = *indexPointer;

number = (int)trio_to_long(&format[index], &tmpformat, BASE_DECIMAL);
index = (int)(tmpformat - format);
if ((number != 0) && (QUALIFIER_POSITION == format[index++]))
{
    *indexPointer = index;
    /*
     * number is decreased by 1, because n$ starts from 1, whereas
     * the array it is indexing starts from 0.
     */
    return number - 1;
}
#endif
return NO_POSITION;
}

#if TRIO_EXTENSION
/*****
 * TrioFindNamespace
 *
 * Find registered user-defined specifier.
 * The prev argument is used for optimization only.
 */
TRIO_PRIVATE trio_userdef_t *
TrioFindNamespace
TRIO_ARGS2((name, prev),
    TRIO_CONST char *name,
    trio_userdef_t **prev)
{
    trio_userdef_t *def;

    if (internalEnterCriticalRegion)
        (void)internalEnterCriticalRegion(NULL);

    for (def = internalUserDef; def; def = def->next)
    {
        /* Case-sensitive string comparison */
        if (trio_equal_case(def->name, name))
            break;

        if (prev)
            *prev = def;
    }

    if (internalLeaveCriticalRegion)
        (void)internalLeaveCriticalRegion(NULL);

    return def;
}
#endif

```

```

/*****
* TrioPower
*
* Description:
* Calculate pow(base, exponent), where number and exponent are integers.
*/
TRIO_PRIVATE trio_long_double_t
TrioPower
TRIO_ARGS2((number, exponent),
    int number,
    int exponent)
{
    trio_long_double_t result;

    if (number == 10)
    {
        switch (exponent)
        {
            /* Speed up calculation of common cases */
        case 0:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E-1);
            break;
        case 1:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+0);
            break;
        case 2:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+1);
            break;
        case 3:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+2);
            break;
        case 4:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+3);
            break;
        case 5:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+4);
            break;
        case 6:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+5);
            break;
        case 7:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+6);
            break;
        case 8:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+7);
            break;
        case 9:
            result = (trio_long_double_t)number * TRIO_SUFFIX_LONG(1E+8);
            break;
        default:
            result = powl((trio_long_double_t)number,
                (trio_long_double_t)exponent);
            break;
        }
    }
}

```



```

    else
    {
        return powl((trio_long_double_t)number, (trio_long_double_t)exponent);
    }
    return result;
}

/*****
 * TrioLogarithm
 */
TRIO_PRIVATE double
TrioLogarithm
TRIO_ARGS2((number, base),
    double number,
    int base)
{
    double result;

    if (number <= 0.0)
    {
        /* x1C crashes on log(0) */
        result = (number == 0.0) ? trio_ninf() : trio_nan();
    }
    else
    {
        if (base == 10)
        {
            result = log10(number);
        }
        else
        {
            result = log10(number) / log10((double)base);
        }
    }
    return result;
}

/*****
 * TrioLogarithmBase
 */
TRIO_PRIVATE double
TrioLogarithmBase
TRIO_ARGS1((base),
    int base)
{
    switch (base)
    {
        case BASE_BINARY : return 1.0;
        case BASE_OCTAL : return 3.0;
        case BASE_DECIMAL : return 3.321928094887362345;
        case BASE_HEX : return 4.0;
        default : return TrioLogarithm((double)base, 2);
    }
}

```

```

/*****
* TrioParse
*
* Description:
* Parse the format string
*/
TRIO_PRIVATE int
TrioParse
TRIO_ARGS5((type, format, parameters, arglist, argarray),
    int type,
    TRIO_CONST char *format,
    trio_parameter_t *parameters,
    TRIO_VA_LIST_PTR arglist,
    trio_pointer_t *argarray)
{
    /* Count the number of times a parameter is referenced */
    unsigned short usedEntries[MAX_PARAMETERS];
    /* Parameter counters */
    int parameterPosition;
    int currentParam;
    int maxParam = -1;
    /* Utility variables */
    trio_flags_t flags;
    int width;
    int precision;
    int varsize;
    int base;
    int index; /* Index into formatting string */
    int dots; /* Count number of dots in modifier part */
    BOOLEAN_T positional; /* Does the specifier have a positional? */
    BOOLEAN_T gotSticky = FALSE; /* Are there any sticky modifiers at all? */
    /*
     * indices specifies the order in which the parameters must be
     * read from the va_args (this is necessary to handle positionals)
     */
    int indices[MAX_PARAMETERS];
    int pos = 0;
    /* Various variables */
    char ch;
#ifdef TRIO_COMPILER_SUPPORTS_MULTIBYTE
    int charlen;
#endif
    int save_errno;
    int i = -1;
    int num;
    char *tmpformat;

    /* One and only one of arglist and argarray must be used */
    assert((arglist != NULL) ^ (argarray != NULL));

    /*
     * The 'parameters' array is not initialized, but we need to
     * know which entries we have used.
     */

```

```

memset(usedEntries, 0, sizeof(usedEntries));

save_errno = errno;
index = 0;
parameterPosition = 0;
#if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
(void)mblen(NULL, 0);
#endif

while (format[index])
{
#if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
if (! isascii(format[index]))
{
/*
 * Multibyte characters cannot be legal specifiers or
 * modifiers, so we skip over them.
 */
charlen = mblen(&format[index], MB_LEN_MAX);
index += (charlen > 0) ? charlen : 1;
continue; /* while */
}
#endif /* TRIO_COMPILER_SUPPORTS_MULTIBYTE */
if (CHAR_IDENTIFIER == format[index++])
{
if (CHAR_IDENTIFIER == format[index])
{
index++;
continue; /* while */
}

flags = FLAGS_NEW;
dots = 0;
currentParam = TrioGetPosition(format, &index);
positional = (NO_POSITION != currentParam);
if (!positional)
{
/* We have no positional, get the next counter */
currentParam = parameterPosition;
}
if (currentParam >= MAX_PARAMETERS)
{
/* Bail out completely to make the error more obvious */
return TRIO_ERROR_RETURN(TRIO_ETOOMANY, index);
}

if (currentParam > maxParam)
maxParam = currentParam;

/* Default values */
width = NO_WIDTH;
precision = NO_PRECISION;
base = NO_BASE;
varsize = NO_SIZE;

```

```

while (TrioIsQualifier(format[index]))
{
    ch = format[index++];

    switch (ch)
    {
case QUALIFIER_SPACE:
    flags |= FLAGS_SPACE;
    break;

case QUALIFIER_PLUS:
    flags |= FLAGS_SHOWSIGN;
    break;

case QUALIFIER_MINUS:
    flags |= FLAGS_LEFTADJUST;
    flags &= ~FLAGS_NILPADDING;
    break;

case QUALIFIER_ALTERNATIVE:
    flags |= FLAGS_ALTERNATIVE;
    break;

case QUALIFIER_DOT:
    if (dots == 0) /* Precision */
    {
        dots++;

        /* Skip if no precision */
        if (QUALIFIER_DOT == format[index])
            break;

        /* After the first dot we have the precision */
        flags |= FLAGS_PRECISION;
        if ((QUALIFIER_STAR == format[index])
#ifdef QUALIFIER_PARAM
            || (QUALIFIER_PARAM == format[index])
#endif
        )
        {
            index++;
            flags |= FLAGS_PRECISION_PARAMETER;

            precision = TrioGetPosition(format, &index);
            if (precision == NO_POSITION)
            {
                parameterPosition++;
                if (positional)
                    precision = parameterPosition;
            }
            else
            {
                precision = currentParam;
                currentParam = precision + 1;
            }
        }
    }
}

```

```

    }
    else
    {
        if (!positional)
            currentParam = precision + 1;
        if (width > maxParam)
            maxParam = precision;
    }
    if (currentParam > maxParam)
        maxParam = currentParam;
}
else
{
    precision = trio_to_long(&format[index],
        &tmpformat,
        BASE_DECIMAL);
    index = (int)(tmpformat - format);
}
}
else if (dots == 1) /* Base */
{
    dots++;

    /* After the second dot we have the base */
    flags |= FLAGS_BASE;
    if ((QUALIFIER_STAR == format[index])
#ifdef QUALIFIER_PARAM
        || (QUALIFIER_PARAM == format[index])
#endif
    )
    {
        index++;
        flags |= FLAGS_BASE_PARAMETER;
        base = TrioGetPosition(format, &index);
        if (base == NO_POSITION)
        {
            parameterPosition++;
            if (positional)
                base = parameterPosition;
            else
            {
                base = currentParam;
                currentParam = base + 1;
            }
        }
    }
    else
    {
        if (!positional)
            currentParam = base + 1;
        if (base > maxParam)
            maxParam = base;
    }
    if (currentParam > maxParam)
        maxParam = currentParam;
}

```

```

    }
    else
    {
        base = trio_to_long(&format[index],
            &tmpformat,
            BASE_DECIMAL);
        if (base > MAX_BASE)
            return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
        index = (int)(tmpformat - format);
    }
    }
    else
    {
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
    }
    break; /* QUALIFIER_DOT */

#ifdef QUALIFIER_PARAM
case QUALIFIER_PARAM:
    type = TYPE_PRINT;
    /* FALLTHROUGH */
#endif
case QUALIFIER_STAR:
    /* This has different meanings for print and scan */
    if (TYPE_PRINT == type)
    {
        /* Read with from parameter */
        flags |= (FLAGS_WIDTH | FLAGS_WIDTH_PARAMETER);
        width = TrioGetPosition(format, &index);
        if (width == NO_POSITION)
        {
            parameterPosition++;
            if (positional)
                width = parameterPosition;
            else
            {
                width = currentParam;
                currentParam = width + 1;
            }
        }
    }
    else
    {
        if (! positional)
            currentParam = width + 1;
        if (width > maxParam)
            maxParam = width;
    }
    if (currentParam > maxParam)
        maxParam = currentParam;
    }
    else
    {
        /* Scan, but do not store result */
        flags |= FLAGS_IGNORE;
    }
}

```

```
break; /* QUALIFIER_STAR */

case '0':
    if (! (flags & FLAGS_LEFTADJUST))
        flags |= FLAGS_NILPADDING;
    /* FALLTHROUGH */
case '1': case '2': case '3': case '4':
case '5': case '6': case '7': case '8': case '9':
    flags |= FLAGS_WIDTH;
    /* &format[index - 1] is used to "rewind" the read
     * character from format
     */
    width = trio_to_long(&format[index - 1],
        &tmpformat,
        BASE_DECIMAL);
    index = (int)(tmpformat - format);
    break;

case QUALIFIER_SHORT:
    if (flags & FLAGS_SHORTEST)
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
    else if (flags & FLAGS_SHORT)
        flags |= FLAGS_SHORTEST;
    else
        flags |= FLAGS_SHORT;
    break;

case QUALIFIER_LONG:
    if (flags & FLAGS_QUAD)
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
    else if (flags & FLAGS_LONG)
        flags |= FLAGS_QUAD;
    else
        flags |= FLAGS_LONG;
    break;

case QUALIFIER_LONG_UPPER:
    flags |= FLAGS_LONGDOUBLE;
    break;

#if defined(QUALIFIER_SIZE_T)
case QUALIFIER_SIZE_T:
    flags |= FLAGS_SIZE_T;
    /* Modify flags for later truncation of number */
    if (sizeof(size_t) == sizeof(trio_ulonglong_t))
        flags |= FLAGS_QUAD;
    else if (sizeof(size_t) == sizeof(long))
        flags |= FLAGS_LONG;
    break;
#endif

#if defined(QUALIFIER_PTRDIFF_T)
case QUALIFIER_PTRDIFF_T:
```

```

    flags |= FLAGS_PTRDIFF_T;
    if (sizeof(ptrdiff_t) == sizeof(trio_ulonglong_t))
        flags |= FLAGS_QUAD;
    else if (sizeof(ptrdiff_t) == sizeof(long))
        flags |= FLAGS_LONG;
    break;
#endif

#ifdef QUALIFIER_INTMAX_T
case QUALIFIER_INTMAX_T:
    flags |= FLAGS_INTMAX_T;
    if (sizeof(trio_intmax_t) == sizeof(trio_ulonglong_t))
        flags |= FLAGS_QUAD;
    else if (sizeof(trio_intmax_t) == sizeof(long))
        flags |= FLAGS_LONG;
    break;
#endif

#ifdef QUALIFIER_QUAD
case QUALIFIER_QUAD:
    flags |= FLAGS_QUAD;
    break;
#endif

#ifdef QUALIFIER_FIXED_SIZE
case QUALIFIER_FIXED_SIZE:
    if (flags & FLAGS_FIXED_SIZE)
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);

    if (flags & (FLAGS_ALL_SIZES | FLAGS_LONGDOUBLE |
        FLAGS_WIDECHAR | FLAGS_VARSIZE_PARAMETER))
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);

    if ((format[index] == '6') &&
        (format[index + 1] == '4'))
    {
        varsize = sizeof(trio_int64_t);
        index += 2;
    }
    else if ((format[index] == '3') &&
        (format[index + 1] == '2'))
    {
        varsize = sizeof(trio_int32_t);
        index += 2;
    }
    else if ((format[index] == '1') &&
        (format[index + 1] == '6'))
    {
        varsize = sizeof(trio_int16_t);
        index += 2;
    }
    else if (format[index] == '8')
    {
        varsize = sizeof(trio_int8_t);
        index++;
    }

```



```
    }
    else
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);

    flags |= FLAGS_FIXED_SIZE;
    break;
#endif

#if defined(QUALIFIER_WIDECHAR)
case QUALIFIER_WIDECHAR:
    flags |= FLAGS_WIDECHAR;
    break;
#endif

#if defined(QUALIFIER_SIZE_T_UPPER)
case QUALIFIER_SIZE_T_UPPER:
    break;
#endif

#if defined(QUALIFIER_QUOTE)
case QUALIFIER_QUOTE:
    flags |= FLAGS_QUOTE;
    break;
#endif

#if defined(QUALIFIER_STICKY)
case QUALIFIER_STICKY:
    flags |= FLAGS_STICKY;
    gotSticky = TRUE;
    break;
#endif

#if defined(QUALIFIER_VARSIZE)
case QUALIFIER_VARSIZE:
    flags |= FLAGS_VARSIZE_PARAMETER;
    parameterPosition++;
    if (positional)
        varsize = parameterPosition;
    else
    {
        varsize = currentParam;
        currentParam = varsize + 1;
    }
    if (currentParam > maxParam)
        maxParam = currentParam;
    break;
#endif

#if defined(QUALIFIER_ROUNDING_UPPER)
case QUALIFIER_ROUNDING_UPPER:
    flags |= FLAGS_ROUNDING;
    break;
#endif
```

```

default:
    /* Bail out completely to make the error more obvious */
    return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
}
} /* while qualifier */

/*
 * Parameters only need the type and value. The value is
 * read later.
 */
if (flags & FLAGS_WIDTH_PARAMETER)
{
    usedEntries[width] += 1;
    parameters[pos].type = FORMAT_PARAMETER;
    parameters[pos].flags = 0;
    indices[width] = pos;
    width = pos++;
}
if (flags & FLAGS_PRECISION_PARAMETER)
{
    usedEntries[precision] += 1;
    parameters[pos].type = FORMAT_PARAMETER;
    parameters[pos].flags = 0;
    indices[precision] = pos;
    precision = pos++;
}
if (flags & FLAGS_BASE_PARAMETER)
{
    usedEntries[base] += 1;
    parameters[pos].type = FORMAT_PARAMETER;
    parameters[pos].flags = 0;
    indices[base] = pos;
    base = pos++;
}
if (flags & FLAGS_VARSIZE_PARAMETER)
{
    usedEntries[varsize] += 1;
    parameters[pos].type = FORMAT_PARAMETER;
    parameters[pos].flags = 0;
    indices[varsize] = pos;
    varsize = pos++;
}

indices[currentParam] = pos;

switch (format[index++])
{
    #if defined(SPECIFIER_CHAR_UPPER)
    case SPECIFIER_CHAR_UPPER:
        flags |= FLAGS_WIDECHAR;
        /* FALLTHROUGH */
    #endif
    case SPECIFIER_CHAR:
        if (flags & FLAGS_LONG)
            flags |= FLAGS_WIDECHAR;

```

```
        else if (flags & FLAGS_SHORT)
        flags &= ~FLAGS_WIDECHAR;
        parameters[pos].type = FORMAT_CHAR;
        break;

#ifdef SPECIFIER_STRING_UPPER
        case SPECIFIER_STRING_UPPER:
            flags |= FLAGS_WIDECHAR;
            /* FALLTHROUGH */
#endif
        case SPECIFIER_STRING:
            if (flags & FLAGS_LONG)
            flags |= FLAGS_WIDECHAR;
            else if (flags & FLAGS_SHORT)
            flags &= ~FLAGS_WIDECHAR;
            parameters[pos].type = FORMAT_STRING;
            break;

        case SPECIFIER_GROUP:
            if (TYPE_SCAN == type)
            {
                int depth = 1;
                parameters[pos].type = FORMAT_GROUP;
                if (format[index] == QUALIFIER_CIRCUMFLEX)
                    index++;
                if (format[index] == SPECIFIER_UNGROUP)
                    index++;
                if (format[index] == QUALIFIER_MINUS)
                    index++;
                /* Skip nested brackets */
                while (format[index] != NIL)
                {
                    if (format[index] == SPECIFIER_GROUP)
                    {
                        depth++;
                    }
                    else if (format[index] == SPECIFIER_UNGROUP)
                    {
                        if (--depth <= 0)
                        {
                            index++;
                            break;
                        }
                    }
                }
                index++;
            }
            break;

        case SPECIFIER_INTEGER:
            parameters[pos].type = FORMAT_INT;
            break;

        case SPECIFIER_UNSIGNED:
```

```
    flags |= FLAGS_UNSIGNED;
    parameters[pos].type = FORMAT_INT;
    break;

case SPECIFIER_DECIMAL:
    /* Disable base modifier */
    flags &= ~FLAGS_BASE_PARAMETER;
    base = BASE_DECIMAL;
    parameters[pos].type = FORMAT_INT;
    break;

case SPECIFIER_OCTAL:
    flags |= FLAGS_UNSIGNED;
    flags &= ~FLAGS_BASE_PARAMETER;
    base = BASE_OCTAL;
    parameters[pos].type = FORMAT_INT;
    break;

#if defined(SPECIFIER_BINARY)
case SPECIFIER_BINARY_UPPER:
    flags |= FLAGS_UPPER;
    /* FALLTHROUGH */
case SPECIFIER_BINARY:
    flags |= FLAGS_NILPADDING;
    flags &= ~FLAGS_BASE_PARAMETER;
    base = BASE_BINARY;
    parameters[pos].type = FORMAT_INT;
    break;
#endif

case SPECIFIER_HEX_UPPER:
    flags |= FLAGS_UPPER;
    /* FALLTHROUGH */
case SPECIFIER_HEX:
    flags |= FLAGS_UNSIGNED;
    flags &= ~FLAGS_BASE_PARAMETER;
    base = BASE_HEX;
    parameters[pos].type = FORMAT_INT;
    break;

case SPECIFIER_FLOAT_E_UPPER:
    flags |= FLAGS_UPPER;
    /* FALLTHROUGH */
case SPECIFIER_FLOAT_E:
    flags |= FLAGS_FLOAT_E;
    parameters[pos].type = FORMAT_DOUBLE;
    break;

case SPECIFIER_FLOAT_G_UPPER:
    flags |= FLAGS_UPPER;
    /* FALLTHROUGH */
case SPECIFIER_FLOAT_G:
    flags |= FLAGS_FLOAT_G;
    parameters[pos].type = FORMAT_DOUBLE;
    break;
```

```

case SPECIFIER_FLOAT_F_UPPER:
    flags |= FLAGS_UPPER;
    /* FALLTHROUGH */
case SPECIFIER_FLOAT_F:
    parameters[pos].type = FORMAT_DOUBLE;
    break;

case SPECIFIER_POINTER:
    if (sizeof(trio_pointer_t) == sizeof(trio_ulonglong_t))
        flags |= FLAGS_QUAD;
    else if (sizeof(trio_pointer_t) == sizeof(long))
        flags |= FLAGS_LONG;
    parameters[pos].type = FORMAT_POINTER;
    break;

case SPECIFIER_COUNT:
    parameters[pos].type = FORMAT_COUNT;
    break;

#if defined(SPECIFIER_HEXFLOAT)
# if defined(SPECIFIER_HEXFLOAT_UPPER)
    case SPECIFIER_HEXFLOAT_UPPER:
        flags |= FLAGS_UPPER;
        /* FALLTHROUGH */
# endif
    case SPECIFIER_HEXFLOAT:
        base = BASE_HEX;
        parameters[pos].type = FORMAT_DOUBLE;
        break;
#endif

#if defined(FORMAT_ERRNO)
    case SPECIFIER_ERRNO:
        parameters[pos].type = FORMAT_ERRNO;
        break;
#endif

#if defined(SPECIFIER_USER_DEFINED_BEGIN)
    case SPECIFIER_USER_DEFINED_BEGIN:
        {
            unsigned int max;
            int without_namespace = TRUE;

            parameters[pos].type = FORMAT_USER_DEFINED;
            parameters[pos].user_name[0] = NIL;
            tmpformat = (char *)&format[index];

            while ((ch = format[index]))
            {
                index++;
                if (ch == SPECIFIER_USER_DEFINED_END)
                {
                    if (without_namespace)

```

```

{
    /* We must get the handle first */
    parameters[pos].type = FORMAT_PARAMETER;
    parameters[pos].indexAfterSpecifier = index;
    parameters[pos].flags = FLAGS_USER_DEFINED;
    /* Adjust parameters for insertion of new one */
    pos++;
    usedEntries[currentParam] += 1;
    parameters[pos].type = FORMAT_USER_DEFINED;
    currentParam++;
    indices[currentParam] = pos;
    if (currentParam > maxParam)
        maxParam = currentParam;
}
/* Copy the user data */
max = (unsigned int)(&format[index] - tmpformat);
if (max > MAX_USER_DATA)
    max = MAX_USER_DATA;
trio_copy_max(parameters[pos].user_data,
    max,
    tmpformat);
break; /* while */
}
if (ch == SPECIFIER_USER_DEFINED_SEPARATOR)
{
    without_namespace = FALSE;
    /* Copy the namespace for later looking-up */
    max = (int)(&format[index] - tmpformat);
    if (max > MAX_USER_NAME)
        max = MAX_USER_NAME;
    trio_copy_max(parameters[pos].user_name,
        max,
        tmpformat);
    tmpformat = (char *)&format[index];
}
}
if (ch != SPECIFIER_USER_DEFINED_END)
    return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
}
break;
#endif /* defined(SPECIFIER_USER_DEFINED_BEGIN) */

default:
    /* Bail out completely to make the error more obvious */
    return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
}

/* Count the number of times this entry has been used */
usedEntries[currentParam] += 1;

/* Find last sticky parameters */
if (gotSticky && !(flags & FLAGS_STICKY))
{
    for (i = pos - 1; i >= 0; i--)
    {

```

```

    if (parameters[i].type == FORMAT_PARAMETER)
        continue;
    if ((parameters[i].flags & FLAGS_STICKY) &&
        (parameters[i].type == parameters[pos].type))
    {
        /* Do not overwrite current qualifiers */
        flags |= (parameters[i].flags & (unsigned long)~FLAGS_STICKY);
        if (width == NO_WIDTH)
            width = parameters[i].width;
        if (precision == NO_PRECISION)
            precision = parameters[i].precision;
        if (base == NO_BASE)
            base = parameters[i].base;
        break;
    }
}

parameters[pos].indexAfterSpecifier = index;
parameters[pos].flags = flags;
parameters[pos].width = width;
parameters[pos].precision = precision;
parameters[pos].base = (base == NO_BASE) ? BASE_DECIMAL : base;
parameters[pos].varsize = varsize;
pos++;

if (! positional)
    parameterPosition++;

} /* if identifier */

} /* while format characters left */

for (num = 0; num <= maxParam; num++)
{
    if (usedEntries[num] != 1)
    {
        if (usedEntries[num] == 0) /* gap detected */
            return TRIO_ERROR_RETURN(TRIO_EGAP, num);
        else /* double references detected */
            return TRIO_ERROR_RETURN(TRIO_EDBLREF, num);
    }

    i = indices[num];

    /*
     * FORMAT_PARAMETERS are only present if they must be read,
     * so it makes no sense to check the ignore flag (besides,
     * the flags variable is not set for that particular type)
     */
    if ((parameters[i].type != FORMAT_PARAMETER) &&
        (parameters[i].flags & FLAGS_IGNORE))
        continue; /* for all arguments */

```

```

/*
 * The stack arguments are read according to ANSI C89
 * default argument promotions:
 *
 * char      = int
 * short     = int
 * unsigned char = unsigned int
 * unsigned short = unsigned int
 * float     = double
 *
 * In addition to the ANSI C89 these types are read (the
 * default argument promotions of C99 has not been
 * considered yet)
 *
 * long long
 * long double
 * size_t
 * ptrdiff_t
 * intmax_t
 */
switch (parameters[i].type)
{
case FORMAT_GROUP:
case FORMAT_STRING:
#ifdef TRIO_WIDECHAR
    if (flags & FLAGS_WIDECHAR)
    {
        parameters[i].data.wstring = (argarray == NULL)
? va_arg(TRIO_VA_LIST_DEREF(arglist), trio_wchar_t *)
: (trio_wchar_t *)(argarray[num]);
    }
    else
#endif
    {
        parameters[i].data.string = (argarray == NULL)
? va_arg(TRIO_VA_LIST_DEREF(arglist), char *)
: (char *) (argarray[num]);
    }
    break;

#ifdef FORMAT_USER_DEFINED
case FORMAT_USER_DEFINED:
#endif
case FORMAT_POINTER:
case FORMAT_COUNT:
case FORMAT_UNKNOWN:
    parameters[i].data.pointer = (argarray == NULL)
? va_arg(TRIO_VA_LIST_DEREF(arglist), trio_pointer_t)
: argarray[num];
    break;

case FORMAT_CHAR:
case FORMAT_INT:
    if (TYPE_SCAN == type)
    {

```



```

        if (argarray == NULL)
            parameters[i].data.pointer =
                (trio_pointer_t)va_arg(TRIO_VA_LIST_DEREF(arglist), trio_pointer_t);
        else
        {
            if (parameters[i].type == FORMAT_CHAR)
                parameters[i].data.pointer =
                    (trio_pointer_t)((char *)argarray[num]);
            else if (parameters[i].flags & FLAGS_SHORT)
                parameters[i].data.pointer =
                    (trio_pointer_t)((short *)argarray[num]);
            else
                parameters[i].data.pointer =
                    (trio_pointer_t)((int *)argarray[num]);
        }
    }
    else
    {
#ifdef QUALIFIER_VARSIZE || defined(QUALIFIER_FIXED_SIZE)
        if (parameters[i].flags
            & (FLAGS_VARSIZE_PARAMETER | FLAGS_FIXED_SIZE))
        {
            if (parameters[i].flags & FLAGS_VARSIZE_PARAMETER)
            {
                /*
                 * Variable sizes are mapped onto the fixed sizes, in
                 * accordance with integer promotion.
                 *
                 * Please note that this may not be portable, as we
                 * only guess the size, not the layout of the numbers.
                 * For example, if int is little-endian, and long is
                 * big-endian, then this will fail.
                 */
                varsize = (int)parameters[parameters[i].varsize].data.number.as_unsigned;
            }
            else
            {
                /* Used for the I<bits> modifiers */
                varsize = parameters[i].varsize;
            }
            parameters[i].flags &= ~FLAGS_ALL_VARSIZES;

            if (varsize <= (int)sizeof(int))
                ;
            else if (varsize <= (int)sizeof(long))
                parameters[i].flags |= FLAGS_LONG;
#ifdef QUALIFIER_INTMAX_T
            else if (varsize <= (int)sizeof(trio_longlong_t))
                parameters[i].flags |= FLAGS_QUAD;
            else
                parameters[i].flags |= FLAGS_INTMAX_T;
#else
            else
                parameters[i].flags |= FLAGS_QUAD;

```

```

#endif
}
#endif /* defined(QUALIFIER_VARSIZE) */
#if defined(QUALIFIER_SIZE_T) || defined(QUALIFIER_SIZE_T_UPPER)
    if (parameters[i].flags & FLAGS_SIZE_T)
        parameters[i].data.number.as_unsigned = (argarray == NULL)
            ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), size_t)
            : (trio_uintmax_t)((size_t *)argarray[num]);
    else
#endif
#if defined(QUALIFIER_PTRDIFF_T)
    if (parameters[i].flags & FLAGS_PTRDIFF_T)
        parameters[i].data.number.as_unsigned = (argarray == NULL)
            ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), ptrdiff_t)
            : (trio_uintmax_t)((ptrdiff_t *)argarray[num]);
    else
#endif
#if defined(QUALIFIER_INTMAX_T)
    if (parameters[i].flags & FLAGS_INTMAX_T)
        parameters[i].data.number.as_unsigned = (argarray == NULL)
            ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), trio_intmax_t)
            : (trio_uintmax_t)((trio_intmax_t *)argarray[num]);
    else
#endif
    if (parameters[i].flags & FLAGS_QUAD)
        parameters[i].data.number.as_unsigned = (argarray == NULL)
            ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), trio_ulonglong_t)
            : (trio_uintmax_t)((trio_ulonglong_t *)argarray[num]);
    else if (parameters[i].flags & FLAGS_LONG)
        parameters[i].data.number.as_unsigned = (argarray == NULL)
            ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), long)
            : (trio_uintmax_t)((long *)argarray[num]);
    else
    {
        if (argarray == NULL)
            parameters[i].data.number.as_unsigned = (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), int);
        else
        {
            if (parameters[i].type == FORMAT_CHAR)
                parameters[i].data.number.as_unsigned = (trio_uintmax_t)((char *)argarray[num]);
            else if (parameters[i].flags & FLAGS_SHORT)
                parameters[i].data.number.as_unsigned = (trio_uintmax_t)((short *)argarray[num]);
            else
                parameters[i].data.number.as_unsigned = (trio_uintmax_t)((int *)argarray[num]);
        }
    }
}
break;

case FORMAT_PARAMETER:
    /*
     * The parameter for the user-defined specifier is a pointer,
     * whereas the rest (width, precision, base) uses an integer.
     */
    if (parameters[i].flags & FLAGS_USER_DEFINED)

```

```

        parameters[i].data.pointer = (argarray == NULL)
        ? va_arg(TRIO_VA_LIST_DEREF(arglist), trio_pointer_t)
        : argarray[num];
    else
        parameters[i].data.number.as_unsigned = (argarray == NULL)
        ? (trio_uintmax_t)va_arg(TRIO_VA_LIST_DEREF(arglist), int)
        : (trio_uintmax_t)((int *)argarray[num]);
    break;

case FORMAT_DOUBLE:
    if (TYPE_SCAN == type)
    {
        if (parameters[i].flags & FLAGS_LONGDOUBLE)
            parameters[i].data.longdoublePointer = (argarray == NULL)
            ? va_arg(TRIO_VA_LIST_DEREF(arglist), trio_long_double_t *)
            : (trio_long_double_t *)argarray[num];
        else
        {
            if (parameters[i].flags & FLAGS_LONG)
                parameters[i].data.doublePointer = (argarray == NULL)
                ? va_arg(TRIO_VA_LIST_DEREF(arglist), double *)
                : (double *)argarray[num];
            else
                parameters[i].data.doublePointer = (argarray == NULL)
                ? (double *)va_arg(TRIO_VA_LIST_DEREF(arglist), float *)
                : (double *)((float *)argarray[num]);
        }
    }
    else
    {
        if (parameters[i].flags & FLAGS_LONGDOUBLE)
            parameters[i].data.longdoubleNumber = (argarray == NULL)
            ? va_arg(TRIO_VA_LIST_DEREF(arglist), trio_long_double_t)
            : (trio_long_double_t)((trio_long_double_t *)argarray[num]);
        else
        {
            if (argarray == NULL)
                parameters[i].data.longdoubleNumber =
                (trio_long_double_t)va_arg(TRIO_VA_LIST_DEREF(arglist), double);
            else
            {
                if (parameters[i].flags & FLAGS_SHORT)
                    parameters[i].data.longdoubleNumber =
                    (trio_long_double_t)((float *)argarray[num]);
                else
                    parameters[i].data.longdoubleNumber =
                    (trio_long_double_t)((double *)argarray[num]);
            }
        }
    }
    break;

#ifdef FORMAT_ERRNO
case FORMAT_ERRNO:

```

```

        parameters[i].data.errorNumber = save_errno;
        break;
    #endif

default:
    break;
}
    } /* for all specifiers */
    return num;
}

/*****
 *
 * FORMATTING
 *
 *****/

/*****
 * TrioWriteNumber
 *
 * Description:
 *   Output a number.
 *   The complexity of this function is a result of the complexity
 *   of the dependencies of the flags.
 */
TRIO_PRIVATE void
TrioWriteNumber
TRIO_ARGS6((self, number, flags, width, precision, base),
    trio_class_t *self,
    trio_uintmax_t number,
    trio_flags_t flags,
    int width,
    int precision,
    int base)
{
    BOOLEAN_T isNegative;
    BOOLEAN_T isNumberZero;
    BOOLEAN_T isPrecisionZero;
    BOOLEAN_T ignoreNumber;
    char buffer[MAX_CHARS_IN(trio_uintmax_t) * (1 + MAX_LOCALE_SEPARATOR_LENGTH) + 1];
    char *bufferend;
    char *pointer;
    TRIO_CONST char *digits;
    int i;
    int length;
    char *p;
    int count;

    assert(VALID(self));
    assert(VALID(self->OutStream));
    assert(((base >= MIN_BASE) && (base <= MAX_BASE)) || (base == NO_BASE));

    digits = (flags & FLAGS_UPPER) ? internalDigitsUpper : internalDigitsLower;

```

```

    if (base == NO_BASE)
        base = BASE_DECIMAL;

    isNumberZero = (number == 0);
    isPrecisionZero = (precision == 0);
    ignoreNumber = (isNumberZero
    && isPrecisionZero
    && !((flags & FLAGS_ALTERNATIVE) && (base == BASE_OCTAL)));

    if (flags & FLAGS_UNSIGNED)
    {
        isNegative = FALSE;
        flags &= ~FLAGS_SHOWSIGN;
    }
    else
    {
        isNegative = ((trio_intmax_t)number < 0);
        if (isNegative)
            number = -((trio_intmax_t)number);
    }

    if (flags & FLAGS_QUAD)
        number &= (trio_ulonglong_t)-1;
    else if (flags & FLAGS_LONG)
        number &= (unsigned long)-1;
    else
        number &= (unsigned int)-1;

    /* Build number */
    pointer = bufferend = &buffer[sizeof(buffer) - 1];
    *pointer-- = NIL;
    for (i = 1; i < (int)sizeof(buffer); i++)
    {
        *pointer-- = digits[number % base];
        number /= base;
        if (number == 0)
            break;

        if ((flags & FLAGS_QUOTE) && TrioFollowedBySeparator(i + 1))
        {
            /*
             * We are building the number from the least significant
             * to the most significant digit, so we have to copy the
             * thousand separator backwards
             */
            length = internalThousandSeparatorLength;
            if (((int)(pointer - buffer) - length) > 0)
            {
                p = &internalThousandSeparator[length - 1];
                while (length-- > 0)
                    *pointer-- = *p--;
            }
        }
    }

```

```

if (! ignoreNumber)
{
    /* Adjust width */
    width -= (bufferend - pointer) - 1;
}

/* Adjust precision */
if (NO_PRECISION != precision)
{
    precision -= (bufferend - pointer) - 1;
    if (precision < 0)
precision = 0;
    flags |= FLAGS_NILPADDING;
}

/* Calculate padding */
count = (! ((flags & FLAGS_LEFTADJUST) || (precision == NO_PRECISION)))
? precision
: 0;

/* Adjust width further */
if (isNegative || (flags & FLAGS_SHOWSIGN) || (flags & FLAGS_SPACE))
width--;
if ((flags & FLAGS_ALTERNATIVE) && !isNumberZero)
{
    switch (base)
    {
case BASE_BINARY:
case BASE_HEX:
width -= 2;
break;
case BASE_OCTAL:
if (!(flags & FLAGS_NILPADDING) || (count == 0))
width--;
break;
default:
break;
}
}

/* Output prefixes spaces if needed */
if (! ((flags & FLAGS_LEFTADJUST) ||
((flags & FLAGS_NILPADDING) && (precision == NO_PRECISION))))
{
    while (width-- > count)
self->OutStream(self, CHAR_ADJUST);
}

/* width has been adjusted for signs and alternatives */
if (isNegative)
self->OutStream(self, '-');
else if (flags & FLAGS_SHOWSIGN)
self->OutStream(self, '+');
else if (flags & FLAGS_SPACE)

```

```

        self->OutStream(self, ' ');

        /* Prefix is not written when the value is zero */
        if ((flags & FLAGS_ALTERNATIVE) && !isNumberZero)
        {
            switch (base)
            {
            case BASE_BINARY:
                self->OutStream(self, '0');
                self->OutStream(self, (flags & FLAGS_UPPER) ? 'B' : 'b');
                break;

            case BASE_OCTAL:
                if (!(flags & FLAGS_NILPADDING) || (count == 0))
                    self->OutStream(self, '0');
                break;

            case BASE_HEX:
                self->OutStream(self, '0');
                self->OutStream(self, (flags & FLAGS_UPPER) ? 'X' : 'x');
                break;

            default:
                break;
        } /* switch base */
    }

    /* Output prefixed zero padding if needed */
    if (flags & FLAGS_NILPADDING)
    {
        if (precision == NO_PRECISION)
            precision = width;
        while (precision-- > 0)
        {
            self->OutStream(self, '0');
            width--;
        }
    }

    if (!ignoreNumber)
    {
        /* Output the number itself */
        while (*(++pointer))
        {
            self->OutStream(self, *pointer);
        }
    }

    /* Output trailing spaces if needed */
    if (flags & FLAGS_LEFTADJUST)
    {
        while (width-- > 0)
            self->OutStream(self, CHAR_ADJUST);
    }

```

```

}

/*****
 * TrioWriteStringCharacter
 *
 * Description:
 *   Output a single character of a string
 */
TRIO_PRIVATE void
TrioWriteStringCharacter
TRIO_ARGS3((self, ch, flags),
  trio_class_t *self,
  int ch,
  trio_flags_t flags)
{
  if (flags & FLAGS_ALTERNATIVE)
  {
    if (! isprint(ch))
    {
      /*
       * Non-printable characters are converted to C escapes or
       * \number, if no C escape exists.
       */
      self->OutStream(self, CHAR_BACKSLASH);
      switch (ch)
      {
        case '007': self->OutStream(self, 'a'); break;
        case 'b': self->OutStream(self, 'b'); break;
        case 'f': self->OutStream(self, 'f'); break;
        case 'n': self->OutStream(self, 'n'); break;
        case 'r': self->OutStream(self, 'r'); break;
        case 't': self->OutStream(self, 't'); break;
        case 'v': self->OutStream(self, 'v'); break;
        case '\\': self->OutStream(self, '\\'); break;
        default:
          self->OutStream(self, 'x');
          TrioWriteNumber(self, (trio_uintmax_t)ch,
            FLAGS_UNSIGNED | FLAGS_NILPADDING,
            2, 2, BASE_HEX);
          break;
      }
    }
    else if (ch == CHAR_BACKSLASH)
    {
      self->OutStream(self, CHAR_BACKSLASH);
      self->OutStream(self, CHAR_BACKSLASH);
    }
    else
    {
      self->OutStream(self, ch);
    }
  }
  else
  {
    self->OutStream(self, ch);
  }
}

```



```

    }
}

/*****
 * TrioWriteString
 *
 * Description:
 * Output a string
 */
TRIO_PRIVATE void
TrioWriteString
TRIO_ARGS5((self, string, flags, width, precision),
    trio_class_t *self,
    TRIO_CONST char *string,
    trio_flags_t flags,
    int width,
    int precision)
{
    int length;
    int ch;

    assert(VALID(self));
    assert(VALID(self->OutStream));

    if (string == NULL)
    {
        string = internalNullString;
        length = sizeof(internalNullString) - 1;
        /* Disable quoting for the null pointer */
        flags &= (~FLAGS_QUOTE);
        width = 0;
    }
    else
    {
        length = trio_length(string);
    }
    if ((NO_PRECISION != precision) &&
        (precision < length))
    {
        length = precision;
    }
    width -= length;

    if (flags & FLAGS_QUOTE)
        self->OutStream(self, CHAR_QUOTE);

    if (! (flags & FLAGS_LEFTADJUST))
    {
        while (width-- > 0)
            self->OutStream(self, CHAR_ADJUST);
    }

    while (length-- > 0)
    {

```

```

    /* The ctype parameters must be an unsigned char (or EOF) */
    ch = (int)((unsigned char)(*string++));
    TrioWriteStringCharacter(self, ch, flags);
}

if (flags & FLAGS_LEFTADJUST)
{
    while (width-- > 0)
self->OutStream(self, CHAR_ADJUST);
}
if (flags & FLAGS_QUOTE)
    self->OutStream(self, CHAR_QUOTE);
}

/*****
* TrioWriteWideStringCharacter
*
* Description:
*   Output a wide string as a multi-byte sequence
*/
#if TRIO_WIDECHAR
TRIO_PRIVATE int
TrioWriteWideStringCharacter
TRIO_ARGS4((self, wch, flags, width),
    trio_class_t *self,
    trio_wchar_t wch,
    trio_flags_t flags,
    int width)
{
    int size;
    int i;
    int ch;
    char *string;
    char buffer[MB_LEN_MAX + 1];

    if (width == NO_WIDTH)
        width = sizeof(buffer);

    size = wctomb(buffer, wch);
    if ((size <= 0) || (size > width) || (buffer[0] == NIL))
        return 0;

    string = buffer;
    i = size;
    while ((width >= i) && (width-- > 0) && (i-- > 0))
    {
        /* The ctype parameters must be an unsigned char (or EOF) */
        ch = (int)((unsigned char)(*string++));
        TrioWriteStringCharacter(self, ch, flags);
    }
    return size;
}
#endif /* TRIO_WIDECHAR */

/*****

```

```
* TrioWriteWideString
*
* Description:
* Output a wide character string as a multi-byte string
*/
#if TRIO_WIDECHAR
TRIO_PRIVATE void
TrioWriteWideString
TRIO_ARGS5((self, wstring, flags, width, precision),
    trio_class_t *self,
    TRIO_CONST trio_wchar_t *wstring,
    trio_flags_t flags,
    int width,
    int precision)
{
    int length;
    int size;

    assert(VALID(self));
    assert(VALID(self->OutStream));

#if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
    (void)mblen(NULL, 0);
#endif

    if (wstring == NULL)
    {
        TrioWriteString(self, NULL, flags, width, precision);
        return;
    }

    if (NO_PRECISION == precision)
    {
        length = INT_MAX;
    }
    else
    {
        length = precision;
        width -= length;
    }

    if (flags & FLAGS_QUOTE)
        self->OutStream(self, CHAR_QUOTE);

    if (! (flags & FLAGS_LEFTADJUST))
    {
        while (width-- > 0)
            self->OutStream(self, CHAR_ADJUST);
    }

    while (length > 0)
    {
        size = TrioWriteWideStringCharacter(self, *wstring++, flags, length);
        if (size == 0)

```

```

break; /* while */
    length -= size;
}

if (flags & FLAGS_LEFTADJUST)
{
    while (width-- > 0)
self->OutStream(self, CHAR_ADJUST);
}
if (flags & FLAGS_QUOTE)
    self->OutStream(self, CHAR_QUOTE);
}
#endif /* TRIO_WIDECHAR */

/*****
* TrioWriteDouble
*
* http://wwwold.dkuug.dk/JTC1/SC22/WG14/www/docs/dr_211.htm
*
* "5.2.4.2.2 paragraph #4
*
* The accuracy [...] is implementation defined, as is the accuracy
* of the conversion between floating-point internal representations
* and string representations performed by the library routine in
* <stdio.h>"
*/
/* FIXME: handle all instances of constant long-double number (L)
* and *l() math functions.
*/
TRIO_PRIVATE void
TrioWriteDouble
TRIO_ARGS6((self, number, flags, width, precision, base),
    trio_class_t *self,
    trio_long_double_t number,
    trio_flags_t flags,
    int width,
    int precision,
    int base)
{
    trio_long_double_t integerNumber;
    trio_long_double_t fractionNumber;
    trio_long_double_t workNumber;
    int integerDigits;
    int fractionDigits;
    int exponentDigits;
    int baseDigits;
    int integerThreshold;
    int fractionThreshold;
    int expectedWidth;
    int exponent = 0;
    unsigned int uExponent = 0;
    int exponentBase;
    trio_long_double_t dblBase;
    trio_long_double_t dblIntegerBase;
    trio_long_double_t dblFractionBase;

```

```

trio_long_double_t integerAdjust;
trio_long_double_t fractionAdjust;
BOOLEAN_T isNegative;
BOOLEAN_T isExponentNegative = FALSE;
BOOLEAN_T requireTwoDigitExponent;
BOOLEAN_T isHex;
TRIO_CONST char *digits;
char *groupingPointer;
int i;
int index;
BOOLEAN_T hasOnlyZeroes;
int zeroes = 0;
register int trailingZeroes;
BOOLEAN_T keepTrailingZeroes;
BOOLEAN_T keepDecimalPoint;
trio_long_double_t epsilon;

assert(VALID(self));
assert(VALID(self->OutStream));
assert((((base >= MIN_BASE) && (base <= MAX_BASE)) || (base == NO_BASE)));

/* Determine sign and look for special quantities */
switch (trio_fpclassify_and_signbit(number, &isNegative))
{
case TRIO_FP_NAN:
    TrioWriteString(self,
        (flags & FLAGS_UPPER)
        ? NAN_UPPER
        : NAN_LOWER,
        flags, width, precision);
    return;

case TRIO_FP_INFINITE:
    if (isNegative)
    {
        /* Negative infinity */
        TrioWriteString(self,
            (flags & FLAGS_UPPER)
            ? "-" INFINITE_UPPER
            : "-" INFINITE_LOWER,
            flags, width, precision);
        return;
    }
    else
    {
        /* Positive infinity */
        TrioWriteString(self,
            (flags & FLAGS_UPPER)
            ? INFINITE_UPPER
            : INFINITE_LOWER,
            flags, width, precision);
        return;
    }
}

```

```

    default:
        /* Finitude */
        break;
    }

    /* Normal numbers */
    if (flags & FLAGS_LONGDOUBLE)
    {
        baseDigits = (base == 10)
? LDBL_DIG
: (int)floor(LDBL_MANT_DIG / TrioLogarithmBase(base));
        epsilon = LDBL_EPSILON;
    }
    else if (flags & FLAGS_SHORT)
    {
        baseDigits = (base == BASE_DECIMAL)
? FLT_DIG
: (int)floor(FLT_MANT_DIG / TrioLogarithmBase(base));
        epsilon = FLT_EPSILON;
    }
    else
    {
        baseDigits = (base == BASE_DECIMAL)
? DBL_DIG
: (int)floor(DBL_MANT_DIG / TrioLogarithmBase(base));
        epsilon = DBL_EPSILON;
    }

    digits = (flags & FLAGS_UPPER) ? internalDigitsUpper : internalDigitsLower;
    isHex = (base == BASE_HEX);
    if (base == NO_BASE)
        base = BASE_DECIMAL;
    dblBase = (trio_long_double_t)base;
    keepTrailingZeroes = !( (flags & FLAGS_ROUNDING) ||
( (flags & FLAGS_FLOAT_G) &&
!(flags & FLAGS_ALTERNATIVE) ) );

    if (flags & FLAGS_ROUNDING)
        precision = baseDigits;

    if (precision == NO_PRECISION)
    {
        if (isHex)
        {
            keepTrailingZeroes = FALSE;
            precision = FLT_MANT_DIG;
        }
        else
        {
            precision = FLT_DIG;
        }
    }

    if (isNegative)
        number = -number;

```

```

if (isHex)
    flags |= FLAGS_FLOAT_E;

if (flags & FLAGS_FLOAT_G)
{
    if (precision == 0)
precision = 1;

    if ((number < 1.0E-4) || (number > powl(base,
        (trio_long_double_t)precision)))
    {
        /* Use scientific notation */
        flags |= FLAGS_FLOAT_E;
    }
    else if (number < 1.0)
    {
        /*
         * Use normal notation. If the integer part of the number is
         * zero, then adjust the precision to include leading fractional
         * zeros.
         */
        workNumber = TrioLogarithm(number, base);
        workNumber = TRIO_FABS(workNumber);
        if (workNumber - floorl(workNumber) < 0.001)
            workNumber--;
        zeroes = (int)floorl(workNumber);
    }
}

if (flags & FLAGS_FLOAT_E)
{
    /* Scale the number */
    workNumber = TrioLogarithm(number, base);
    if (trio_isinf(workNumber) == -1)
    {
        exponent = 0;
        /* Undo setting */
        if (flags & FLAGS_FLOAT_G)
            flags &= ~FLAGS_FLOAT_E;
    }
    else
    {
        exponent = (int)floorl(workNumber);
        number /= powl(dblBase, (trio_long_double_t)exponent);
        isExponentNegative = (exponent < 0);
        uExponent = (isExponentNegative) ? -exponent : exponent;
        if (isHex)
            uExponent *= 4; /* log16(2) */
        /* No thousand separators */
        flags &= ~FLAGS_QUOTE;
    }
}

```

```

integerNumber = floorl(number);
fractionNumber = number - integerNumber;

/*
 * Truncated number.
 *
 * Precision is number of significant digits for FLOAT_G
 * and number of fractional digits for others.
 */
integerDigits = (integerNumber > epsilon)
    ? 1 + (int)TrioLogarithm(integerNumber, base)
    : 1;
fractionDigits = ((flags & FLAGS_FLOAT_G) && (zeroes == 0))
    ? precision - integerDigits
    : zeroes + precision;

dblFractionBase = TrioPower(base, fractionDigits);

workNumber = number + 0.5 / dblFractionBase;
if (floorl(number) != floorl(workNumber))
{
    if (flags & FLAGS_FLOAT_E)
    {
        /* Adjust if number was rounded up one digit (ie. 0.99 to 1.00) */
        exponent++;
        isExponentNegative = (exponent < 0);
        uExponent = (isExponentNegative) ? -exponent : exponent;
        if (isHex)
            uExponent *= 4; /* log16(2) */
        workNumber = (number + 0.5 / dblFractionBase) / dblBase;
        integerNumber = floorl(workNumber);
        fractionNumber = workNumber - integerNumber;
    }
    else
    {
        /* Adjust if number was rounded up one digit (ie. 99 to 100) */
        integerNumber = floorl(number + 0.5);
        fractionNumber = 0.0;
        integerDigits = (integerNumber > epsilon)
            ? 1 + (int)TrioLogarithm(integerNumber, base)
            : 1;
    }
}

/* Estimate accuracy */
integerAdjust = fractionAdjust = 0.5;
if (flags & FLAGS_ROUNDING)
{
    if (integerDigits > baseDigits)
    {
        integerThreshold = baseDigits;
        fractionDigits = 0;
        dblFractionBase = 1.0;
        fractionThreshold = 0;
        precision = 0; /* Disable decimal-point */
    }
}

```



```

integerAdjust = TrioPower(base, integerDigits - integerThreshold - 1);
fractionAdjust = 0.0;
}
else
{
integerThreshold = integerDigits;
fractionThreshold = fractionDigits - integerThreshold;
fractionAdjust = 1.0;
}
}
else
{
integerThreshold = INT_MAX;
fractionThreshold = INT_MAX;
}

/*
 * Calculate expected width.
 * sign + integer part + thousands separators + decimal point
 * + fraction + exponent
 */
fractionAdjust /= dblFractionBase;
hasOnlyZeroes = (floorl((fractionNumber + fractionAdjust) * dblFractionBase) < epsilon);
keepDecimalPoint = ( (flags & FLAGS_ALTERNATIVE) ||
    !((precision == 0) ||
    (!keepTrailingZeroes && hasOnlyZeroes)) );
if (flags & FLAGS_FLOAT_E)
{
exponentDigits = (uExponent == 0)
? 1
: (int)ceil(TrioLogarithm((double)(uExponent + 1),
(isHex) ? 10.0 : base));
}
else
exponentDigits = 0;
requireTwoDigitExponent = ((base == BASE_DECIMAL) && (exponentDigits == 1));

expectedWidth = integerDigits + fractionDigits
+ (keepDecimalPoint
? internalDecimalPointLength
: 0)
+ ((flags & FLAGS_QUOTE)
? TrioCalcThousandSeparatorLength(integerDigits)
: 0);
if (isNegative || (flags & FLAGS_SHOWSIGN) || (flags & FLAGS_SPACE))
expectedWidth += sizeof("-") - 1;
if (exponentDigits > 0)
expectedWidth += exponentDigits +
((requireTwoDigitExponent ? sizeof("E+0") : sizeof("E+")) - 1);
if (isHex)
expectedWidth += sizeof("0X") - 1;

/* Output prefixing */
if (flags & FLAGS_NILPADDING)

```

```

        {
            /* Leading zeros must be after sign */
            if (isNegative)
                self->OutStream(self, '-');
            else if (flags & FLAGS_SHOWSIGN)
                self->OutStream(self, '+');
            else if (flags & FLAGS_SPACE)
                self->OutStream(self, ' ');
            if (isHex)
            {
                self->OutStream(self, '0');
                self->OutStream(self, (flags & FLAGS_UPPER) ? 'X' : 'x');
            }
            if (!(flags & FLAGS_LEFTADJUST))
            {
                for (i = expectedWidth; i < width; i++)
                {
                    self->OutStream(self, '0');
                }
            }
            else
            {
                /* Leading spaces must be before sign */
                if (!(flags & FLAGS_LEFTADJUST))
                {
                    for (i = expectedWidth; i < width; i++)
                    {
                        self->OutStream(self, CHAR_ADJUST);
                    }
                }
                if (isNegative)
                    self->OutStream(self, '-');
                else if (flags & FLAGS_SHOWSIGN)
                    self->OutStream(self, '+');
                else if (flags & FLAGS_SPACE)
                    self->OutStream(self, ' ');
                if (isHex)
                {
                    self->OutStream(self, '0');
                    self->OutStream(self, (flags & FLAGS_UPPER) ? 'X' : 'x');
                }
            }

            /* Output the integer part and thousand separators */
            dblIntegerBase = 1.0 / TrioPower(base, integerDigits - 1);
            for (i = 0; i < integerDigits; i++)
            {
                workNumber = floorl(((integerNumber + integerAdjust) * dblIntegerBase));
                if (i > integerThreshold)
                {
                    /* Beyond accuracy */
                    self->OutStream(self, digits[0]);
                }
            }
            else

```

```

{
    self->OutStream(self, digits[(int)fmodl(workNumber, dblBase)]);
}

    dblIntegerBase *= dblBase;

    if (((flags & (FLAGS_FLOAT_E | FLAGS_QUOTE)) == FLAGS_QUOTE)
    && TrioFollowedBySeparator(integerDigits - i))
    {
        for (groupingPointer = internalThousandSeparator;
            *groupingPointer != NIL;
            groupingPointer++)
        {
            self->OutStream(self, *groupingPointer);
        }
    }
}

/* Insert decimal point and build the fraction part */
trailingZeroes = 0;

if (keepDecimalPoint)
{
    if (internalDecimalPoint)
    {
        self->OutStream(self, internalDecimalPoint);
    }
    else
    {
        for (i = 0; i < internalDecimalPointLength; i++)
        {
            self->OutStream(self, internalDecimalPointString[i]);
        }
    }
}

for (i = 0; i < fractionDigits; i++)
{
    if ((integerDigits > integerThreshold) || (i > fractionThreshold))
    {
        /* Beyond accuracy */
        trailingZeroes++;
    }
    else
    {
        fractionNumber *= dblBase;
        fractionAdjust *= dblBase;
        workNumber = floorl(fractionNumber + fractionAdjust);
        fractionNumber -= workNumber;
        index = (int)fmodl(workNumber, dblBase);
        if (index == 0)
        {
            trailingZeroes++;
        }
    }
    else

```

```

        {
            while (trailingZeroes > 0)
            {
                /* Not trailing zeroes after all */
                self->OutStream(self, digits[0]);
                trailingZeroes--;
            }
            self->OutStream(self, digits[index]);
        }
    }

    if (keepTrailingZeroes)
    {
        while (trailingZeroes > 0)
        {
            self->OutStream(self, digits[0]);
            trailingZeroes--;
        }
    }

    /* Output exponent */
    if (exponentDigits > 0)
    {
        self->OutStream(self,
            isHex
            ? ((flags & FLAGS_UPPER) ? 'P' : 'p')
            : ((flags & FLAGS_UPPER) ? 'E' : 'e'));
        self->OutStream(self, (isExponentNegative) ? '-' : '+');

        /* The exponent must contain at least two digits */
        if (requireTwoDigitExponent)
            self->OutStream(self, '0');

        if (isHex)
            base = 16.0;
        else
            base = 10.0;
        exponentBase = (int)TrioPower(base, exponentDigits - 1);
        for (i = 0; i < exponentDigits; i++)
        {
            self->OutStream(self, digits[(uExponent / exponentBase) % base]);
            exponentBase /= base;
        }
    }

    /* Output trailing spaces */
    if (flags & FLAGS_LEFTADJUST)
    {
        for (i = expectedWidth; i < width; i++)
        {
            self->OutStream(self, CHAR_ADJUST);
        }
    }
}

/*****
 * TrioFormatProcess

```

```

*
* Description:
* This is the main engine for formatting output
*/
TRIO_PRIVATE int
TrioFormatProcess
TRIO_ARGS3((data, format, parameters),
    trio_class_t *data,
    TRIO_CONST char *format,
    trio_parameter_t *parameters)
{
#if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
    int charlen;
#endif
    int i;
    TRIO_CONST char *string;
    trio_pointer_t pointer;
    trio_flags_t flags;
    int width;
    int precision;
    int base;
    int index;

    index = 0;
    i = 0;
#if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
    (void)mblen(NULL, 0);
#endif

    while (format[index])
    {
        #if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
            if (! isascii(format[index]))
            {
                charlen = mblen(&format[index], MB_LEN_MAX);
                /*
                 * Only valid multibyte characters are handled here. Invalid
                 * multibyte characters (charlen == -1) are handled as normal
                 * characters.
                 */
                if (charlen != -1)
                {
                    while (charlen-- > 0)
                    {
                        data->OutStream(data, format[index++]);
                    }
                    continue; /* while characters left in formatting string */
                }
            }
        #endif /* TRIO_COMPILER_SUPPORTS_MULTIBYTE */
        if (CHAR_IDENTIFIER == format[index])
        {
            if (CHAR_IDENTIFIER == format[index + 1])
            {

```

```

        data->OutStream(data, CHAR_IDENTIFIER);
        index += 2;
    }
    else
    {
        /* Skip the parameter entries */
        while (parameters[i].type == FORMAT_PARAMETER)
            i++;

        flags = parameters[i].flags;

        /* Find width */
        width = parameters[i].width;
        if (flags & FLAGS_WIDTH_PARAMETER)
        {
            /* Get width from parameter list */
            width = (int)parameters[width].data.number.as_signed;
            if (width < 0)
            {
                /*
                 * A negative width is the same as the - flag and
                 * a positive width.
                 */
                flags |= FLAGS_LEFTADJUST;
                flags &= ~FLAGS_NILPADDING;
                width = -width;
            }
        }

        /* Find precision */
        if (flags & FLAGS_PRECISION)
        {
            precision = parameters[i].precision;
            if (flags & FLAGS_PRECISION_PARAMETER)
            {
                /* Get precision from parameter list */
                precision = (int)parameters[precision].data.number.as_signed;
                if (precision < 0)
                {
                    /*
                     * A negative precision is the same as no
                     * precision
                     */
                    precision = NO_PRECISION;
                }
            }
        }
        else
        {
            precision = NO_PRECISION;
        }

        /* Find base */
        base = parameters[i].base;
        if (flags & FLAGS_BASE_PARAMETER)

```

```
{
    /* Get base from parameter list */
    base = (int)parameters[base].data.number.as_signed;
}

    switch (parameters[i].type)
    {
case FORMAT_CHAR:
    if (flags & FLAGS_QUOTE)
        data->OutStream(data, CHAR_QUOTE);
    if (! (flags & FLAGS_LEFTADJUST))
    {
        while (--width > 0)
            data->OutStream(data, CHAR_ADJUST);
    }
#ifdef TRIO_WIDECHAR
    if (flags & FLAGS_WIDECHAR)
    {
        TrioWriteWideStringCharacter(data,
            (trio_wchar_t)parameters[i].data.number.as_signed,
            flags,
            NO_WIDTH);
    }
    else
#endif
    {
        TrioWriteStringCharacter(data,
            (int)parameters[i].data.number.as_signed,
            flags);
    }

    if (flags & FLAGS_LEFTADJUST)
    {
        while(--width > 0)
            data->OutStream(data, CHAR_ADJUST);
    }
    if (flags & FLAGS_QUOTE)
        data->OutStream(data, CHAR_QUOTE);

    break; /* FORMAT_CHAR */

case FORMAT_INT:
    TrioWriteNumber(data,
        parameters[i].data.number.as_unsigned,
        flags,
        width,
        precision,
        base);

    break; /* FORMAT_INT */

case FORMAT_DOUBLE:
    TrioWriteDouble(data,
        parameters[i].data.longdoubleNumber,
```

```

    flags,
    width,
    precision,
    base);
    break; /* FORMAT_DOUBLE */

case FORMAT_STRING:
#ifdef TRIO_WIDECHAR
    if (flags & FLAGS_WIDECHAR)
    {
        TrioWriteWideString(data,
        parameters[i].data.wstring,
        flags,
        width,
        precision);
    }
    else
#endif
    {
        TrioWriteString(data,
        parameters[i].data.string,
        flags,
        width,
        precision);
    }
    break; /* FORMAT_STRING */

case FORMAT_POINTER:
{
    trio_reference_t reference;

    reference.data = data;
    reference.parameter = &parameters[i];
    trio_print_pointer(&reference, parameters[i].data.pointer);
}
break; /* FORMAT_POINTER */

case FORMAT_COUNT:
    pointer = parameters[i].data.pointer;
    if (NULL != pointer)
    {
        /*
         * C99 paragraph 7.19.6.1.8 says "the number of
         * characters written to the output stream so far by
         * this call", which is data->committed
         */
#ifdef QUALIFIER_SIZE_T || defined(QUALIFIER_SIZE_T_UPPER)
        if (flags & FLAGS_SIZE_T)
            *(size_t *)pointer = (size_t)data->committed;
        else
#endif
#ifdef QUALIFIER_PTRDIFF_T
        if (flags & FLAGS_PTRDIFF_T)
            *(ptrdiff_t *)pointer = (ptrdiff_t)data->committed;
        else

```



```
#endif
#if defined(QUALIFIER_INTMAX_T)
    if (flags & FLAGS_INTMAX_T)
        *(trio_intmax_t *)pointer = (trio_intmax_t)data->committed;
    else
#endif
        if (flags & FLAGS_QUAD)
        {
            *(trio_ulonglong_t *)pointer = (trio_ulonglong_t)data->committed;
        }
        else if (flags & FLAGS_LONG)
        {
            *(long int *)pointer = (long int)data->committed;
        }
        else if (flags & FLAGS_SHORT)
        {
            *(short int *)pointer = (short int)data->committed;
        }
        else
        {
            *(int *)pointer = (int)data->committed;
        }
    }
    break; /* FORMAT_COUNT */

case FORMAT_PARAMETER:
    break; /* FORMAT_PARAMETER */

#if defined(FORMAT_ERRNO)
case FORMAT_ERRNO:
    string = trio_error(parameters[i].data.errorNumber);
    if (string)
    {
        TrioWriteString(data,
            string,
            flags,
            width,
            precision);
    }
    else
    {
        data->OutStream(data, '#');
        TrioWriteNumber(data,
            (trio_uintmax_t)parameters[i].data.errorNumber,
            flags,
            width,
            precision,
            BASE_DECIMAL);
    }
    break; /* FORMAT_ERRNO */
#endif /* defined(FORMAT_ERRNO) */

#if defined(FORMAT_USER_DEFINED)
case FORMAT_USER_DEFINED:
```

```

    {
        trio_reference_t reference;
        trio_userdef_t *def = NULL;

        if (parameters[i].user_name[0] == NIL)
        {
            /* Use handle */
            if ((i > 0) ||
                (parameters[i - 1].type == FORMAT_PARAMETER))
                def = (trio_userdef_t *)parameters[i - 1].data.pointer;
            else
            {
                /* Look up namespace */
                def = TrioFindNamespace(parameters[i].user_name, NULL);
            }
            if (def) {
                reference.data = data;
                reference.parameter = &parameters[i];
                def->callback(&reference);
            }
        }
        break;
    }
    #endif /* defined(FORMAT_USER_DEFINED) */

default:
    break;
} /* switch parameter type */

    /* Prepare for next */
    index = parameters[i].indexAfterSpecifier;
    i++;
}
else /* not identifier */
{
    data->OutStream(data, format[index++]);
}
return data->processed;
}

/*****
 * TrioFormatRef
 */
TRIO_PRIVATE int
TrioFormatRef
TRIO_ARGS4((reference, format, arglist, argarray),
    trio_reference_t *reference,
    TRIO_CONST char *format,
    TRIO_VA_LIST_PTR arglist,
    trio_pointer_t *argarray)
{
    int status;
    trio_parameter_t parameters[MAX_PARAMETERS];

```

```

status = TrioParse(TYPE_PRINT, format, parameters, arglist, argarray);
if (status < 0)
    return status;

status = TrioFormatProcess(reference->data, format, parameters);
if (reference->data->error != 0)
{
    status = reference->data->error;
}
return status;
}

/*****
* TrioFormat
*/
TRIO_PRIVATE int
TrioFormat
TRIO_ARGS6((destination, destinationSize, OutStream, format, arglist, argarray),
    trio_pointer_t destination,
    size_t destinationSize,
    void (*OutStream) TRIO_PROTO((trio_class_t *, int)),
    TRIO_CONST char *format,
    TRIO_VA_LIST_PTR arglist,
    trio_pointer_t *argarray)
{
    int status;
    trio_class_t data;
    trio_parameter_t parameters[MAX_PARAMETERS];

    assert(VALID(OutStream));
    assert(VALID(format));

    memset(&data, 0, sizeof(data));
    data.OutStream = OutStream;
    data.location = destination;
    data.max = destinationSize;
    data.error = 0;

#ifdef USE_LOCALE
    if (NULL == internalLocaleValues)
    {
        TrioSetLocale();
    }
#endif

    status = TrioParse(TYPE_PRINT, format, parameters, arglist, argarray);
    if (status < 0)
        return status;

    status = TrioFormatProcess(&data, format, parameters);
    if (data.error != 0)
    {
        status = data.error;
    }
}

```

```

    }
    return status;
}

/*****
 * TrioOutputStreamFile
 */
TRIO_PRIVATE void
TrioOutputStreamFile
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    FILE *file;

    assert(VALID(self));
    assert(VALID(self->location));

    file = (FILE *)self->location;
    self->processed++;
    if (fputc(output, file) == EOF)
    {
        self->error = TRIO_ERROR_RETURN(TRIO_EOF, 0);
    }
    else
    {
        self->committed++;
    }
}

/*****
 * TrioOutputStreamFileDescriptor
 */
TRIO_PRIVATE void
TrioOutputStreamFileDescriptor
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    int fd;
    char ch;

    assert(VALID(self));

    fd = *((int *)self->location);
    ch = (char)output;
    self->processed++;
    if (write(fd, &ch, sizeof(char)) == -1)
    {
        self->error = TRIO_ERROR_RETURN(TRIO_ERRNO, 0);
    }
    else
    {
        self->committed++;
    }
}

```

```

}

/*****
 * TrioOutputStreamCustom
 */
TRIO_PRIVATE void
TrioOutputStreamCustom
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    int status;
    trio_custom_t *data;

    assert(VALID(self));
    assert(VALID(self->location));

    data = (trio_custom_t *)self->location;
    if (data->stream.out)
    {
        status = (data->stream.out)(data->closure, output);
        if (status >= 0)
        {
            self->committed++;
        }
        else
        {
            if (self->error == 0)
            {
                self->error = TRIO_ERROR_RETURN(TRIO_ECUSTOM, -status);
            }
        }
    }
    self->processed++;
}

/*****
 * TrioOutputStreamString
 */
TRIO_PRIVATE void
TrioOutputStreamString
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    char **buffer;

    assert(VALID(self));
    assert(VALID(self->location));

    buffer = (char **)self->location;
    **buffer = (char)output;
    (*buffer)++;
    self->processed++;
}

```

```

    self->committed++;
}

/*****
 * TrioOutputStreamStringMax
 */
TRIO_PRIVATE void
TrioOutputStreamStringMax
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    char **buffer;

    assert(VALID(self));
    assert(VALID(self->location));

    buffer = (char **)self->location;

    if (self->processed < self->max)
    {
        **buffer = (char)output;
        (*buffer)++;
        self->committed++;
    }
    self->processed++;
}

/*****
 * TrioOutputStreamStringDynamic
 */
TRIO_PRIVATE void
TrioOutputStreamStringDynamic
TRIO_ARGS2((self, output),
    trio_class_t *self,
    int output)
{
    assert(VALID(self));
    assert(VALID(self->location));

    if (self->error == 0)
    {
        trio_xstring_append_char((trio_string_t *)self->location,
            (char)output);
        self->committed++;
    }
    /* The processed variable must always be increased */
    self->processed++;
}

/*****
 *
 * Formatted printing functions
 *
 *****/

```

```

#if defined(TRIO_DOCUMENTATION)
# include "doc/doc_printf.h"
#endif
/** @addtogroup Printf
    @{
*/

/*****
 * printf
 */

/**
    Print to standard output stream.

    @param format Formatting string.
    @param ... Arguments.
    @return Number of printed characters.
 */
TRIO_PUBLIC int
trio_printf
TRIO_VARGS2((format, va_list),
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioFormat(stdout, 0, TrioOutputStreamFile, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

/**
    Print to standard output stream.

    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
 */
TRIO_PUBLIC int
trio_vprintf
TRIO_ARGS2((format, args),
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(format));

    return TrioFormat(stdout, 0, TrioOutputStreamFile, format, TRIO_VA_LIST_ADDR(args), NULL);
}

```

```

/**
    Print to standard output stream.

    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_printfv
TRIO_ARGS2((format, args),
    TRIO_CONST char *format,
    trio_pointer_t * args)
{
    assert(VALID(format));

    return TrioFormat(stdout, 0, TrioOutputStreamFile, format, NULL, args);
}

/*****
 * fprintf
 */

/**
    Print to file.

    @param file File pointer.
    @param format Formatting string.
    @param ... Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_fprintf
TRIO_VARGS3((file, format, va_alist),
    FILE *file,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(file));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioFormat(file, 0, TrioOutputStreamFile, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

/**
    Print to file.

    @param file File pointer.
    @param format Formatting string.
    @param args Arguments.

```



```

    @return Number of printed characters.
    */
TRIO_PUBLIC int
trio_vfprintf
TRIO_ARGS3((file, format, args),
    FILE *file,
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(file));
    assert(VALID(format));

    return TrioFormat(file, 0, TrioOutputStreamFile, format, TRIO_VA_LIST_ADDR(args), NULL);
}

/**
    Print to file.

    @param file File pointer.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
    */
TRIO_PUBLIC int
trio_fprintfv
TRIO_ARGS3((file, format, args),
    FILE *file,
    TRIO_CONST char *format,
    trio_pointer_t * args)
{
    assert(VALID(file));
    assert(VALID(format));

    return TrioFormat(file, 0, TrioOutputStreamFile, format, NULL, args);
}

/*****
* dprintf
*/

/**
    Print to file descriptor.

    @param fd File descriptor.
    @param format Formatting string.
    @param ... Arguments.
    @return Number of printed characters.
    */
TRIO_PUBLIC int
trio_dprintf
TRIO_VARS3((fd, format, va_alist),
    int fd,
    TRIO_CONST char *format,
    TRIO_VA_DECL)

```

```

{
    int status;
    va_list args;

    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioFormat(&fd, 0, TrioOutputStreamFileDescriptor, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

/**
    Print to file descriptor.

    @param fd File descriptor.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_vdprintf
TRIO_ARGS3((fd, format, args),
    int fd,
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(format));

    return TrioFormat(&fd, 0, TrioOutputStreamFileDescriptor, format, TRIO_VA_LIST_ADDR(args), NULL);
}

/**
    Print to file descriptor.

    @param fd File descriptor.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_dprintfv
TRIO_ARGS3((fd, format, args),
    int fd,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    assert(VALID(format));

    return TrioFormat(&fd, 0, TrioOutputStreamFileDescriptor, format, NULL, args);
}

/*****
* cprintf
*/

```

```
TRIO_PUBLIC int
trio_cprintf
TRIO_VARGS4((stream, closure, format, va_alist),
    trio_outstream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    data.stream.out = stream;
    data.closure = closure;
    status = TrioFormat(&data, 0, TrioOutputStreamCustom, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vprintf
TRIO_ARGS4((stream, closure, format, args),
    trio_outstream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    va_list args)
{
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));

    data.stream.out = stream;
    data.closure = closure;
    return TrioFormat(&data, 0, TrioOutputStreamCustom, format, TRIO_VA_LIST_ADDR(args), NULL);
}

TRIO_PUBLIC int
trio_cprintfv
TRIO_ARGS4((stream, closure, format, args),
    trio_outstream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    void **args)
{
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));
```

```

    data.stream.out = stream;
    data.closure = closure;
    return TrioFormat(&data, 0, TrioOutputStreamCustom, format, NULL, args);
}

/*****
 * sprintf
 */

/**
    Print to string.

    @param buffer Output string.
    @param format Formatting string.
    @param ... Arguments.
    @return Number of printed characters.
 */
TRIO_PUBLIC int
trio_sprintf
TRIO_VARGS3((buffer, format, va_alist),
    char *buffer,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(buffer));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioFormat(&buffer, 0, TrioOutputStreamString, format, TRIO_VA_LIST_ADDR(args), NULL);
    *buffer = NIL; /* Terminate with NIL character */
    TRIO_VA_END(args);
    return status;
}

/**
    Print to string.

    @param buffer Output string.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
 */
TRIO_PUBLIC int
trio_vsprintf
TRIO_ARGS3((buffer, format, args),
    char *buffer,
    TRIO_CONST char *format,
    va_list args)
{
    int status;

    assert(VALID(buffer));

```

```

    assert(VALID(format));

    status = TrioFormat(&buffer, 0, TrioOutputStreamString, format, TRIO_VA_LIST_ADDR(args), NULL);
    *buffer = NIL;
    return status;
}

/**
    Print to string.

    @param buffer Output string.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_sprintfv
TRIO_ARGS3((buffer, format, args),
    char *buffer,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    int status;

    assert(VALID(buffer));
    assert(VALID(format));

    status = TrioFormat(&buffer, 0, TrioOutputStreamString, format, NULL, args);
    *buffer = NIL;
    return status;
}

/*****
* snprintf
*/

/**
    Print at most @p max characters to string.

    @param buffer Output string.
    @param max Maximum number of characters to print.
    @param format Formatting string.
    @param ... Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_snprintf
TRIO_VARGS4((buffer, max, format, va_alist),
    char *buffer,
    size_t max,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;

```

```

    va_list args;

    assert(VALID(buffer));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioFormat(&buffer, max > 0 ? max - 1 : 0,
        TrioOutputStreamMax, format, TRIO_VA_LIST_ADDR(args), NULL);
    if (max > 0)
        *buffer = NIL;
    TRIO_VA_END(args);
    return status;
}

/**
    Print at most @p max characters to string.

    @param buffer Output string.
    @param max Maximum number of characters to print.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_vsnprintf
TRIO_ARGS4((buffer, max, format, args),
    char *buffer,
    size_t max,
    TRIO_CONST char *format,
    va_list args)
{
    int status;

    assert(VALID(buffer));
    assert(VALID(format));

    status = TrioFormat(&buffer, max > 0 ? max - 1 : 0,
        TrioOutputStreamMax, format, TRIO_VA_LIST_ADDR(args), NULL);
    if (max > 0)
        *buffer = NIL;
    return status;
}

/**
    Print at most @p max characters to string.

    @param buffer Output string.
    @param max Maximum number of characters to print.
    @param format Formatting string.
    @param args Arguments.
    @return Number of printed characters.
*/
TRIO_PUBLIC int
trio_snprintfv
TRIO_ARGS4((buffer, max, format, args),

```

```

    char *buffer,
    size_t max,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    int status;

    assert(VALID(buffer));
    assert(VALID(format));

    status = TrioFormat(&buffer, max > 0 ? max - 1 : 0,
        TrioOutputStreamStringMax, format, NULL, args);
    if (max > 0)
        *buffer = NIL;
    return status;
}

/*****
 * snprintfcat
 * Appends the new string to the buffer string overwriting the '\0'
 * character at the end of buffer.
 */
TRIO_PUBLIC int
trio_snprintfcat
TRIO_VARGS4((buffer, max, format, va_alist),
    char *buffer,
    size_t max,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;
    size_t buf_len;

    TRIO_VA_START(args, format);

    assert(VALID(buffer));
    assert(VALID(format));

    buf_len = trio_length(buffer);
    buffer = &buffer[buf_len];

    status = TrioFormat(&buffer, max - 1 - buf_len,
        TrioOutputStreamStringMax, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    *buffer = NIL;
    return status;
}

TRIO_PUBLIC int
trio_vsnprintfcat
TRIO_ARGS4((buffer, max, format, args),
    char *buffer,
    size_t max,

```

```

    TRIO_CONST char *format,
    va_list args)
{
    int status;
    size_t buf_len;

    assert(VALID(buffer));
    assert(VALID(format));

    buf_len = trio_length(buffer);
    buffer = &buffer[buf_len];
    status = TrioFormat(&buffer, max - 1 - buf_len,
        TrioOutputStreamStringMax, format, TRIO_VA_LIST_ADDR(args), NULL);
    *buffer = NIL;
    return status;
}

/*****
 * trio_aprintf
 */

/* Deprecated */
TRIO_PUBLIC char *
trio_aprintf
TRIO_VARGS2((format, va_alist),
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    va_list args;
    trio_string_t *info;
    char *result = NULL;

    assert(VALID(format));

    info = trio_xstring_duplicate("");
    if (info)
    {
        TRIO_VA_START(args, format);
        (void)TrioFormat(info, 0, TrioOutputStreamStringDynamic,
            format, TRIO_VA_LIST_ADDR(args), NULL);
        TRIO_VA_END(args);

        trio_string_terminate(info);
        result = trio_string_extract(info);
        trio_string_destroy(info);
    }
    return result;
}

/* Deprecated */
TRIO_PUBLIC char *
trio_vaprintf
TRIO_ARGS2((format, args),
    TRIO_CONST char *format,
    va_list args)

```



```
{
    trio_string_t *info;
    char *result = NULL;

    assert(VALID(format));

    info = trio_xstring_duplicate("");
    if (info)
    {
        (void)TrioFormat(info, 0, TrioOutputStreamStringDynamic,
            format, TRIO_VA_LIST_ADDR(args), NULL);
        trio_string_terminate(info);
        result = trio_string_extract(info);
        trio_string_destroy(info);
    }
    return result;
}

TRIO_PUBLIC int
trio_asprintf
TRIO_VARGS3((result, format, va_alist),
    char **result,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    va_list args;
    int status;
    trio_string_t *info;

    assert(VALID(format));

    *result = NULL;

    info = trio_xstring_duplicate("");
    if (info == NULL)
    {
        status = TRIO_ERROR_RETURN(TRIO_ENOMEM, 0);
    }
    else
    {
        TRIO_VA_START(args, format);
        status = TrioFormat(info, 0, TrioOutputStreamStringDynamic,
            format, TRIO_VA_LIST_ADDR(args), NULL);
        TRIO_VA_END(args);
        if (status >= 0)
        {
            trio_string_terminate(info);
            *result = trio_string_extract(info);
        }
        trio_string_destroy(info);
    }
    return status;
}
```

```

TRIO_PUBLIC int
trio_vasprintf
TRIO_ARGS3((result, format, args),
    char **result,
    TRIO_CONST char *format,
    va_list args)
{
    int status;
    trio_string_t *info;

    assert(VALID(format));

    *result = NULL;

    info = trio_xstring_duplicate("");
    if (info == NULL)
    {
        status = TRIO_ERROR_RETURN(TRIO_ENOMEM, 0);
    }
    else
    {
        status = TrioFormat(info, 0, TrioOutputStreamStringDynamic,
            format, TRIO_VA_LIST_ADDR(args), NULL);
        if (status >= 0)
        {
            trio_string_terminate(info);
            *result = trio_string_extract(info);
        }
        trio_string_destroy(info);
    }
    return status;
}

/** @} End of Printf documentation module */

/*****
 *
 * CALLBACK
 *
 *****/

#if defined(TRIO_DOCUMENTATION)
# include "doc/doc_register.h"
#endif
/**
    @addtogroup UserDefined
    @{
*/

#if TRIO_EXTENSION

/*****
 * trio_register
 */

```

```

/**
    Register new user-defined specifier.

    @param callback
    @param name
    @return Handle.
*/
TRIO_PUBLIC trio_pointer_t
trio_register
TRIO_ARGS2((callback, name),
    trio_callback_t callback,
    TRIO_CONST char *name)
{
    trio_userdef_t *def;
    trio_userdef_t *prev = NULL;

    if (callback == NULL)
        return NULL;

    if (name)
    {
        /* Handle built-in namespaces */
        if (name[0] == ':')
        {
            if (trio_equal(name, ":enter"))
            {
                internalEnterCriticalRegion = callback;
            }
            else if (trio_equal(name, ":leave"))
            {
                internalLeaveCriticalRegion = callback;
            }
            return NULL;
        }

        /* Bail out if namespace is too long */
        if (trio_length(name) >= MAX_USER_NAME)
            return NULL;

        /* Bail out if namespace already is registered */
        def = TrioFindNamespace(name, &prev);
        if (def)
            return NULL;
    }

    def = (trio_userdef_t *)TRIO_MALLOC(sizeof(trio_userdef_t));
    if (def)
    {
        if (internalEnterCriticalRegion)
            (void)internalEnterCriticalRegion(NULL);

        if (name)
        {
            /* Link into internal list */

```

```

    if (prev == NULL)
        internalUserDef = def;
    else
        prev->next = def;
}
/* Initialize */
def->callback = callback;
def->name = (name == NULL)
? NULL
: trio_duplicate(name);
def->next = NULL;

    if (internalLeaveCriticalRegion)
(void)internalLeaveCriticalRegion(NULL);
}
return (trio_pointer_t)def;
}

/**
    Unregister an existing user-defined specifier.

    @param handle
    */
void
trio_unregister
TRIO_ARGS1((handle),
    trio_pointer_t handle)
{
    trio_userdef_t *self = (trio_userdef_t *)handle;
    trio_userdef_t *def;
    trio_userdef_t *prev = NULL;

    assert(VALID(self));

    if (self->name)
    {
        def = TrioFindNamespace(self->name, &prev);
        if (def)
        {
            if (internalEnterCriticalRegion)
                (void)internalEnterCriticalRegion(NULL);

            if (prev == NULL)
                internalUserDef = NULL;
            else
                prev->next = def->next;

            if (internalLeaveCriticalRegion)
                (void)internalLeaveCriticalRegion(NULL);
        }
        trio_destroy(self->name);
    }
    TRIO_FREE(self);
}

```

```

/*****
 * trio_get_format [public]
 */
TRIO_CONST char *
trio_get_format
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
#ifdef FORMAT_USER_DEFINED
    assert(((trio_reference_t *)ref)->parameter->type == FORMAT_USER_DEFINED);
#endif

    return (((trio_reference_t *)ref)->parameter->user_data);
}

/*****
 * trio_get_argument [public]
 */
trio_pointer_t
trio_get_argument
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
#ifdef FORMAT_USER_DEFINED
    assert(((trio_reference_t *)ref)->parameter->type == FORMAT_USER_DEFINED);
#endif

    return ((trio_reference_t *)ref)->parameter->data.pointer;
}

/*****
 * trio_get_width / trio_set_width [public]
 */
int
trio_get_width
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return ((trio_reference_t *)ref)->parameter->width;
}

void
trio_set_width
TRIO_ARGS2((ref, width),
    trio_pointer_t ref,
    int width)
{
    ((trio_reference_t *)ref)->parameter->width = width;
}

/*****
 * trio_get_precision / trio_set_precision [public]
 */
int

```

```

trio_get_precision
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->precision);
}

void
trio_set_precision
TRIO_ARGS2((ref, precision),
    trio_pointer_t ref,
    int precision)
{
    ((trio_reference_t *)ref)->parameter->precision = precision;
}

/*****
 * trio_get_base / trio_set_base [public]
 */
int
trio_get_base
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->base);
}

void
trio_set_base
TRIO_ARGS2((ref, base),
    trio_pointer_t ref,
    int base)
{
    ((trio_reference_t *)ref)->parameter->base = base;
}

/*****
 * trio_get_long / trio_set_long [public]
 */
int
trio_get_long
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_LONG)
        ? TRUE
        : FALSE;
}

void
trio_set_long
TRIO_ARGS2((ref, is_long),
    trio_pointer_t ref,
    int is_long)
{

```

```

    if (is_long)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_LONG;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_LONG;
}

/*****
 * trio_get_longlong / trio_set_longlong [public]
 */
int
trio_get_longlong
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_QUAD)
        ? TRUE
        : FALSE;
}

void
trio_set_longlong
TRIO_ARGS2((ref, is_longlong),
    trio_pointer_t ref,
    int is_longlong)
{
    if (is_longlong)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_QUAD;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_QUAD;
}

/*****
 * trio_get_longdouble / trio_set_longdouble [public]
 */
int
trio_get_longdouble
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_LONGDOUBLE)
        ? TRUE
        : FALSE;
}

void
trio_set_longdouble
TRIO_ARGS2((ref, is_longdouble),
    trio_pointer_t ref,
    int is_longdouble)
{
    if (is_longdouble)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_LONGDOUBLE;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_LONGDOUBLE;
}

```

```

}

/*****
 * trio_get_short / trio_set_short [public]
 */
int
trio_get_short
TRIO_ARGS1((ref),
  trio_pointer_t ref)
{
  return (((trio_reference_t *)ref)->parameter->flags & FLAGS_SHORT)
    ? TRUE
    : FALSE;
}

void
trio_set_short
TRIO_ARGS2((ref, is_short),
  trio_pointer_t ref,
  int is_short)
{
  if (is_short)
    ((trio_reference_t *)ref)->parameter->flags |= FLAGS_SHORT;
  else
    ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_SHORT;
}

/*****
 * trio_get_shortshort / trio_set_shortshort [public]
 */
int
trio_get_shortshort
TRIO_ARGS1((ref),
  trio_pointer_t ref)
{
  return (((trio_reference_t *)ref)->parameter->flags & FLAGS_SHORTSHORT)
    ? TRUE
    : FALSE;
}

void
trio_set_shortshort
TRIO_ARGS2((ref, is_shortshort),
  trio_pointer_t ref,
  int is_shortshort)
{
  if (is_shortshort)
    ((trio_reference_t *)ref)->parameter->flags |= FLAGS_SHORTSHORT;
  else
    ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_SHORTSHORT;
}

/*****
 * trio_get_alternative / trio_set_alternative [public]
 */

```



```

int
trio_get_alternative
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_ALTERNATIVE)
        ? TRUE
        : FALSE;
}

void
trio_set_alternative
TRIO_ARGS2((ref, is_alternative),
    trio_pointer_t ref,
    int is_alternative)
{
    if (is_alternative)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_ALTERNATIVE;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_ALTERNATIVE;
}

/*****
 * trio_get_alignment / trio_set_alignment [public]
 */
int
trio_get_alignment
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_LEFTADJUST)
        ? TRUE
        : FALSE;
}

void
trio_set_alignment
TRIO_ARGS2((ref, is_leftaligned),
    trio_pointer_t ref,
    int is_leftaligned)
{
    if (is_leftaligned)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_LEFTADJUST;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_LEFTADJUST;
}

/*****
 * trio_get_spacing /trio_set_spacing [public]
 */
int
trio_get_spacing
TRIO_ARGS1((ref),
    trio_pointer_t ref)

```

```

{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_SPACE)
        ? TRUE
        : FALSE;
}

void
trio_set_spacing
TRIO_ARGS2((ref, is_space),
    trio_pointer_t ref,
    int is_space)
{
    if (is_space)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_SPACE;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_SPACE;
}

/*****
 * trio_get_sign / trio_set_sign [public]
 */
int
trio_get_sign
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_SHOWSIGN)
        ? TRUE
        : FALSE;
}

void
trio_set_sign
TRIO_ARGS2((ref, is_sign),
    trio_pointer_t ref,
    int is_sign)
{
    if (is_sign)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_SHOWSIGN;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_SHOWSIGN;
}

/*****
 * trio_get_padding / trio_set_padding [public]
 */
int
trio_get_padding
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_NILPADDING)
        ? TRUE
        : FALSE;
}

```

```

void
trio_set_padding
TRIO_ARGS2((ref, is_padding),
    trio_pointer_t ref,
    int is_padding)
{
    if (is_padding)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_NILPADDING;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_NILPADDING;
}

/*****
 * trio_get_quote / trio_set_quote [public]
 */
int
trio_get_quote
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_QUOTE)
        ? TRUE
        : FALSE;
}

void
trio_set_quote
TRIO_ARGS2((ref, is_quote),
    trio_pointer_t ref,
    int is_quote)
{
    if (is_quote)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_QUOTE;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_QUOTE;
}

/*****
 * trio_get_upper / trio_set_upper [public]
 */
int
trio_get_upper
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_UPPER)
        ? TRUE
        : FALSE;
}

void
trio_set_upper
TRIO_ARGS2((ref, is_upper),

```

```

        trio_pointer_t ref,
        int is_upper)
{
    if (is_upper)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_UPPER;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_UPPER;
}

/*****
 * trio_get_largest / trio_set_largest [public]
 */
#ifdef TRIO_C99
int
trio_get_largest
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_INTMAX_T)
        ? TRUE
        : FALSE;
}

void
trio_set_largest
TRIO_ARGS2((ref, is_largest),
    trio_pointer_t ref,
    int is_largest)
{
    if (is_largest)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_INTMAX_T;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_INTMAX_T;
}
#endif

/*****
 * trio_get_ptrdiff / trio_set_ptrdiff [public]
 */
int
trio_get_ptrdiff
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_PTRDIFF_T)
        ? TRUE
        : FALSE;
}

void
trio_set_ptrdiff
TRIO_ARGS2((ref, is_ptrdiff),
    trio_pointer_t ref,
    int is_ptrdiff)
{

```

```

    if (is_ptrdiff)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_PTRDIFF_T;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_PTRDIFF_T;
}

/*****
 * trio_get_size / trio_set_size [public]
 */
#if TRIO_C99
int
trio_get_size
TRIO_ARGS1((ref),
    trio_pointer_t ref)
{
    return (((trio_reference_t *)ref)->parameter->flags & FLAGS_SIZE_T)
        ? TRUE
        : FALSE;
}

void
trio_set_size
TRIO_ARGS2((ref, is_size),
    trio_pointer_t ref,
    int is_size)
{
    if (is_size)
        ((trio_reference_t *)ref)->parameter->flags |= FLAGS_SIZE_T;
    else
        ((trio_reference_t *)ref)->parameter->flags &= ~FLAGS_SIZE_T;
}
#endif

/*****
 * trio_print_int [public]
 */
void
trio_print_int
TRIO_ARGS2((ref, number),
    trio_pointer_t ref,
    int number)
{
    trio_reference_t *self = (trio_reference_t *)ref;

    TrioWriteNumber(self->data,
        (trio_uintmax_t)number,
        self->parameter->flags,
        self->parameter->width,
        self->parameter->precision,
        self->parameter->base);
}

/*****
 * trio_print_uint [public]

```

```

*/
void
trio_print_uint
TRIO_ARGS2((ref, number),
    trio_pointer_t ref,
    unsigned int number)
{
    trio_reference_t *self = (trio_reference_t *)ref;

    TrioWriteNumber(self->data,
        (trio_uintmax_t)number,
        self->parameter->flags | FLAGS_UNSIGNED,
        self->parameter->width,
        self->parameter->precision,
        self->parameter->base);
}

/*****
 * trio_print_double [public]
 */
void
trio_print_double
TRIO_ARGS2((ref, number),
    trio_pointer_t ref,
    double number)
{
    trio_reference_t *self = (trio_reference_t *)ref;

    TrioWriteDouble(self->data,
        number,
        self->parameter->flags,
        self->parameter->width,
        self->parameter->precision,
        self->parameter->base);
}

/*****
 * trio_print_string [public]
 */
void
trio_print_string
TRIO_ARGS2((ref, string),
    trio_pointer_t ref,
    char *string)
{
    trio_reference_t *self = (trio_reference_t *)ref;

    TrioWriteString(self->data,
        string,
        self->parameter->flags,
        self->parameter->width,
        self->parameter->precision);
}

/*****

```

```

* trio_print_ref [public]
*/
int
trio_print_ref
TRIO_VARGS3((ref, format, va_alist),
    trio_pointer_t ref,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list arglist;

    assert(VALID(format));

    TRIO_VA_START(arglist, format);
    status = TrioFormatRef((trio_reference_t *)ref, format, TRIO_VA_LIST_ADDR(arglist), NULL);
    TRIO_VA_END(arglist);
    return status;
}

/*****
* trio_vprint_ref [public]
*/
int
trio_vprint_ref
TRIO_ARGS3((ref, format, arglist),
    trio_pointer_t ref,
    TRIO_CONST char *format,
    va_list arglist)
{
    assert(VALID(format));

    return TrioFormatRef((trio_reference_t *)ref, format, TRIO_VA_LIST_ADDR(arglist), NULL);
}

/*****
* trio_printv_ref [public]
*/
int
trio_printv_ref
TRIO_ARGS3((ref, format, argarray),
    trio_pointer_t ref,
    TRIO_CONST char *format,
    trio_pointer_t *argarray)
{
    assert(VALID(format));

    return TrioFormatRef((trio_reference_t *)ref, format, NULL, argarray);
}

#endif /* TRIO_EXTENSION */

/*****
* trio_print_pointer [public]

```

```

*/
void
trio_print_pointer
TRIO_ARGS2((ref, pointer),
    trio_pointer_t ref,
    trio_pointer_t pointer)
{
    trio_reference_t *self = (trio_reference_t *)ref;
    trio_flags_t flags;
    trio_uintmax_t number;

    if (NULL == pointer)
    {
        TRIO_CONST char *string = internalNullString;
        while (*string)
            self->data->OutStream(self->data, *string++);
    }
    else
    {
        /*
         * The subtraction of the null pointer is a workaround
         * to avoid a compiler warning. The performance overhead
         * is negligible (and likely to be removed by an
         * optimizing compiler). The (char *) casting is done
         * to please ANSI C++.
         */
        number = (trio_uintmax_t)((char *)pointer - (char *)0);
        /* Shrink to size of pointer */
        number &= (trio_uintmax_t)-1;
        flags = self->parameter->flags;
        flags |= (FLAGS_UNSIGNED | FLAGS_ALTERNATIVE |
            FLAGS_NILPADDING);
        TrioWriteNumber(self->data,
            number,
            flags,
            POINTER_WIDTH,
            NO_PRECISION,
            BASE_HEX);
    }
}

/** @} End of UserDefined documentation module */

/*****
*
* LOCALES
*
*****/

/*****
* trio_locale_set_decimal_point
*
* Decimal point can only be one character. The input argument is a
* string to enable multibyte characters. At most MB_LEN_MAX characters
* will be used.

```



```

*/
TRIO_PUBLIC void
trio_locale_set_decimal_point
TRIO_ARGS1((decimalPoint),
    char *decimalPoint)
{
#ifdef USE_LOCALE
    if (NULL == internalLocaleValues)
    {
        TrioSetLocale();
    }
#endif
    internalDecimalPointLength = trio_length(decimalPoint);
    if (internalDecimalPointLength == 1)
    {
        internalDecimalPoint = *decimalPoint;
    }
    else
    {
        internalDecimalPoint = NIL;
        trio_copy_max(internalDecimalPointString,
            sizeof(internalDecimalPointString),
            decimalPoint);
    }
}

/*****
 * trio_locale_set_thousand_separator
 *
 * See trio_locale_set_decimal_point
 */
TRIO_PUBLIC void
trio_locale_set_thousand_separator
TRIO_ARGS1((thousandSeparator),
    char *thousandSeparator)
{
#ifdef USE_LOCALE
    if (NULL == internalLocaleValues)
    {
        TrioSetLocale();
    }
#endif
    trio_copy_max(internalThousandSeparator,
        sizeof(internalThousandSeparator),
        thousandSeparator);
    internalThousandSeparatorLength = trio_length(internalThousandSeparator);
}

/*****
 * trio_locale_set_grouping
 *
 * Array of bytes. Reversed order.
 *
 * CHAR_MAX : No further grouping

```

```

* 0      : Repeat last group for the remaining digits (not necessary
*          as C strings are zero-terminated)
* n      : Set current group to n
*
* Same order as the grouping attribute in LC_NUMERIC.
*/
TRIO_PUBLIC void
trio_locale_set_grouping
TRIO_ARGS1((grouping),
char *grouping)
{
#ifdef USE_LOCALE
    if (NULL == internalLocaleValues)
    {
        TrioSetLocale();
    }
#endif
    trio_copy_max(internalGrouping,
sizeof(internalGrouping),
grouping);
}

/*****
*
* SCANNING
*
*****/

/*****
* TrioSkipWhitespaces
*/
TRIO_PRIVATE int
TrioSkipWhitespaces
TRIO_ARGS1((self),
trio_class_t *self)
{
    int ch;

    ch = self->current;
    while (isspace(ch))
    {
        self->InStream(self, &ch);
    }
    return ch;
}

/*****
* TrioGetCollation
*/
#ifdef TRIO_EXTENSION
TRIO_PRIVATE void
TrioGetCollation(TRIO_NOARGS)
{
    int i;

```

```

int j;
int k;
char first[2];
char second[2];

/* This is computationally expensive */
first[1] = NIL;
second[1] = NIL;
for (i = 0; i < MAX_CHARACTER_CLASS; i++)
{
    k = 0;
    first[0] = (char)i;
    for (j = 0; j < MAX_CHARACTER_CLASS; j++)
    {
        second[0] = (char)j;
        if (trio_equal_locale(first, second))
            internalCollationArray[i][k++] = (char)j;
    }
    internalCollationArray[i][k] = NIL;
}
}
#endif

/*****
* TrioGetCharacterClass
*
* FIXME:
* multibyte
*/
TRIO_PRIVATE int
TrioGetCharacterClass
TRIO_ARGS4((format, indexPointer, flagsPointer, characterclass),
    TRIO_CONST char *format,
    int *indexPointer,
    trio_flags_t *flagsPointer,
    int *characterclass)
{
    int index = *indexPointer;
    int i;
    char ch;
    char range_begin;
    char range_end;

    *flagsPointer &= ~FLAGS_EXCLUDE;

    if (format[index] == QUALIFIER_CIRCUMFLEX)
    {
        *flagsPointer |= FLAGS_EXCLUDE;
        index++;
    }
}
/*
* If the ungroup character is at the beginning of the scanlist,
* it will be part of the class, and a second ungroup character
* must follow to end the group.

```

```

    */
    if (format[index] == SPECIFIER_UNGROUP)
    {
        characterclass[(int)SPECIFIER_UNGROUP]++;
        index++;
    }
    /*
    * Minus is used to specify ranges. To include minus in the class,
    * it must be at the beginning of the list
    */
    if (format[index] == QUALIFIER_MINUS)
    {
        characterclass[(int)QUALIFIER_MINUS]++;
        index++;
    }
    /* Collect characters */
    for (ch = format[index];
        (ch != SPECIFIER_UNGROUP) && (ch != NIL);
        ch = format[++index])
    {
        switch (ch)
        {
        case QUALIFIER_MINUS: /* Scanlist ranges */

            /*
            * Both C99 and UNIX98 describes ranges as implementation-
            * defined.
            *
            * We support the following behaviour (although this may
            * change as we become wiser)
            * - only increasing ranges, ie. [a-b] but not [b-a]
            * - transitive ranges, ie. [a-b-c] == [a-c]
            * - trailing minus, ie. [a-] is interpreted as an 'a'
            *   and a '-'
            * - duplicates (although we can easily convert these
            *   into errors)
            */
            range_begin = format[index - 1];
            range_end = format[++index];
            if (range_end == SPECIFIER_UNGROUP)
            {
                /* Trailing minus is included */
                characterclass[(int)ch]++;
                ch = range_end;
                break; /* for */
            }
            if (range_end == NIL)
                return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
            if (range_begin > range_end)
                return TRIO_ERROR_RETURN(TRIO_ERANGE, index);

            for (i = (int)range_begin; i <= (int)range_end; i++)
                characterclass[i]++;

            ch = range_end;

```

```

    break;

#if TRIO_EXTENSION

case SPECIFIER_GROUP:

    switch (format[index + 1])
    {
        case QUALIFIER_DOT: /* Collating symbol */
            /*
             * FIXME: This will be easier to implement when multibyte
             * characters have been implemented. Until now, we ignore
             * this feature.
             */
            for (i = index + 2; ; i++)
            {
                if (format[i] == NIL)
                    /* Error in syntax */
                    return -1;
                else if (format[i] == QUALIFIER_DOT)
                    break; /* for */
            }
            if (format[++i] != SPECIFIER_UNGROUP)
                return -1;

            index = i;
            break;

        case QUALIFIER_EQUAL: /* Equivalence class expressions */
            {
                unsigned int j;
                unsigned int k;

                if (internalCollationUnconverted)
                {
                    /* Lazy evaluation of collation array */
                    TrioGetCollation();
                    internalCollationUnconverted = FALSE;
                }
                for (i = index + 2; ; i++)
                {
                    if (format[i] == NIL)
                        /* Error in syntax */
                        return -1;
                    else if (format[i] == QUALIFIER_EQUAL)
                        break; /* for */
                    else
                    {
                        /* Mark any equivalent character */
                        k = (unsigned int)format[i];
                        for (j = 0; internalCollationArray[k][j] != NIL; j++)
                            characterclass[(int)internalCollationArray[k][j]]++;
                    }
                }
            }
    }

```

```

if (format[++i] != SPECIFIER_UNGROUP)
    return -1;

index = i;
}
break;

case QUALIFIER_COLON: /* Character class expressions */

    if (trio_equal_max(CLASS_ALNUM, sizeof(CLASS_ALNUM) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (isalnum(i))
                characterclass[i]++;
        index += sizeof(CLASS_ALNUM) - 1;
    }
    else if (trio_equal_max(CLASS_ALPHA, sizeof(CLASS_ALPHA) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (isalpha(i))
                characterclass[i]++;
        index += sizeof(CLASS_ALPHA) - 1;
    }
    else if (trio_equal_max(CLASS_CNTRL, sizeof(CLASS_CNTRL) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (isctrl(i))
                characterclass[i]++;
        index += sizeof(CLASS_CNTRL) - 1;
    }
    else if (trio_equal_max(CLASS_DIGIT, sizeof(CLASS_DIGIT) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (isdigit(i))
                characterclass[i]++;
        index += sizeof(CLASS_DIGIT) - 1;
    }
    else if (trio_equal_max(CLASS_GRAPH, sizeof(CLASS_GRAPH) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (isgraph(i))
                characterclass[i]++;
        index += sizeof(CLASS_GRAPH) - 1;
    }
    else if (trio_equal_max(CLASS_LOWER, sizeof(CLASS_LOWER) - 1,
&format[index]))
    {
        for (i = 0; i < MAX_CHARACTER_CLASS; i++)
            if (islower(i))
                characterclass[i]++;
    }

```

```

    index += sizeof(CLASS_LOWER) - 1;
}
    else if (trio_equal_max(CLASS_PRINT, sizeof(CLASS_PRINT) - 1,
        &format[index]))
{
    for (i = 0; i < MAX_CHARACTER_CLASS; i++)
        if (isprint(i))
            characterclass[i]++;
    index += sizeof(CLASS_PRINT) - 1;
}
    else if (trio_equal_max(CLASS_PUNCT, sizeof(CLASS_PUNCT) - 1,
        &format[index]))
{
    for (i = 0; i < MAX_CHARACTER_CLASS; i++)
        if (ispunct(i))
            characterclass[i]++;
    index += sizeof(CLASS_PUNCT) - 1;
}
    else if (trio_equal_max(CLASS_SPACE, sizeof(CLASS_SPACE) - 1,
        &format[index]))
{
    for (i = 0; i < MAX_CHARACTER_CLASS; i++)
        if (isspace(i))
            characterclass[i]++;
    index += sizeof(CLASS_SPACE) - 1;
}
    else if (trio_equal_max(CLASS_UPPER, sizeof(CLASS_UPPER) - 1,
        &format[index]))
{
    for (i = 0; i < MAX_CHARACTER_CLASS; i++)
        if (isupper(i))
            characterclass[i]++;
    index += sizeof(CLASS_UPPER) - 1;
}
    else if (trio_equal_max(CLASS_XDIGIT, sizeof(CLASS_XDIGIT) - 1,
        &format[index]))
{
    for (i = 0; i < MAX_CHARACTER_CLASS; i++)
        if (isxdigit(i))
            characterclass[i]++;
    index += sizeof(CLASS_XDIGIT) - 1;
}
    else
    {
        characterclass[(int)ch]++;
    }
    break;

default:
    characterclass[(int)ch]++;
    break;
}
break;

```

```

#endif /* TRIO_EXTENSION */

default:
    characterclass[(int)ch]++;
    break;
}
}
return 0;
}

/*****
 * TrioReadNumber
 *
 * We implement our own number conversion in preference of strtol and
 * strtoul, because we must handle 'long long' and thousand separators.
 */
TRIO_PRIVATE BOOLEAN_T
TrioReadNumber
TRIO_ARGS5((self, target, flags, width, base),
    trio_class_t *self,
    trio_uintmax_t *target,
    trio_flags_t flags,
    int width,
    int base)
{
    trio_uintmax_t number = 0;
    int digit;
    int count;
    BOOLEAN_T isNegative = FALSE;
    BOOLEAN_T gotNumber = FALSE;
    int j;

    assert(VALID(self));
    assert(VALID(self->InStream));
    assert((base >= MIN_BASE && base <= MAX_BASE) || (base == NO_BASE));

    if (internalDigitsUnconverted)
    {
        /* Lazy evaluation of digits array */
        memset(internalDigitArray, -1, sizeof(internalDigitArray));
        for (j = 0; j < (int)sizeof(internalDigitsLower) - 1; j++)
        {
            internalDigitArray[(int)internalDigitsLower[j]] = j;
            internalDigitArray[(int)internalDigitsUpper[j]] = j;
        }
        internalDigitsUnconverted = FALSE;
    }

    TrioSkipWhitespaces(self);

    if (!(flags & FLAGS_UNSIGNED))
    {
        /* Leading sign */
        if (self->current == '+')
        {

```



```
        self->InStream(self, NULL);
    }
    else if (self->current == '-')
    {
        self->InStream(self, NULL);
        isNegative = TRUE;
    }
}

count = self->processed;

if (flags & FLAGS_ALTERNATIVE)
{
    switch (base)
    {
    case NO_BASE:
    case BASE_OCTAL:
    case BASE_HEX:
    case BASE_BINARY:
        if (self->current == '0')
        {
            self->InStream(self, NULL);
            if (self->current)
            {
                if ((base == BASE_HEX) &&
                    (trio_to_upper(self->current) == 'X'))
                {
                    self->InStream(self, NULL);
                }
                else if ((base == BASE_BINARY) &&
                    (trio_to_upper(self->current) == 'B'))
                {
                    self->InStream(self, NULL);
                }
            }
        }
        else
            return FALSE;
        break;
    default:
        break;
    }
}

while (((width == NO_WIDTH) || (self->processed - count < width)) &&
(! ((self->current == EOF) || isspace(self->current))))
{
    if (isascii(self->current))
    {
        digit = internalDigitArray[self->current];
        /* Abort if digit is not allowed in the specified base */
        if ((digit == -1) || (digit >= base))
            break;
    }
}
```

```

        else if (flags & FLAGS_QUOTE)
        {
            /* Compare with thousands separator */
            for (j = 0; internalThousandSeparator[j] && self->current; j++)
            {
                if (internalThousandSeparator[j] != self->current)
                    break;

                self->InStream(self, NULL);
            }
            if (internalThousandSeparator[j])
                break; /* Mismatch */
            else
                continue; /* Match */
        }
        else
            break;

        number *= base;
        number += digit;
        gotNumber = TRUE; /* we need at least one digit */

        self->InStream(self, NULL);
    }

    /* Was anything read at all? */
    if (!gotNumber)
        return FALSE;

    if (target)
        *target = (isNegative) ? -((trio_intmax_t)number) : number;
    return TRUE;
}

/*****
 * TrioReadChar
 */
TRIO_PRIVATE int
TrioReadChar
TRIO_ARGS4((self, target, flags, width),
    trio_class_t *self,
    char *target,
    trio_flags_t flags,
    int width)
{
    int i;
    char ch;
    trio_uintmax_t number;

    assert(VALID(self));
    assert(VALID(self->InStream));

    for (i = 0;
        (self->current != EOF) && (i < width);
        i++)

```

```

    {
        ch = (char)self->current;
        self->InStream(self, NULL);
        if ((flags & FLAGS_ALTERNATIVE) && (ch == CHAR_BACKSLASH))
    {
switch (self->current)
    {
        case '\\': ch = '\\'; break;
        case 'a': ch = '\007'; break;
        case 'b': ch = '\b'; break;
        case 'f': ch = '\f'; break;
        case 'n': ch = '\n'; break;
        case 'r': ch = '\r'; break;
        case 't': ch = '\t'; break;
        case 'v': ch = '\v'; break;
        default:
            if (isdigit(self->current))
    {
        /* Read octal number */
        if (!TrioReadNumber(self, &number, 0, 3, BASE_OCTAL))
            return 0;
        ch = (char)number;
    }
        else if (trio_to_upper(self->current) == 'X')
    {
        /* Read hexadecimal number */
        self->InStream(self, NULL);
        if (!TrioReadNumber(self, &number, 0, 2, BASE_HEX))
            return 0;
        ch = (char)number;
    }
        else
    {
        ch = (char)self->current;
    }
        break;
    }
    }

    if (target)
target[i] = ch;
    }
    return i + 1;
}

/*****
 * TrioReadString
 */
TRIO_PRIVATE BOOLEAN_T
TrioReadString
TRIO_ARGS4((self, target, flags, width),
    trio_class_t *self,
    char *target,
    trio_flags_t flags,

```

```

    int width)
{
    int i;

    assert(VALID(self));
    assert(VALID(self->InStream));

    TrioSkipWhitespaces(self);

    /*
     * Continue until end of string is reached, a whitespace is encountered,
     * or width is exceeded
     */
    for (i = 0;
         ((width == NO_WIDTH) || (i < width)) &&
         (! ((self->current == EOF) || isspace(self->current))))
        i++;
    {
        if (TrioReadChar(self, (target ? &target[i] : 0), flags, 1) == 0)
            break; /* for */
    }
    if (target)
        target[i] = NIL;
    return TRUE;
}

/*****
 * TrioReadWideChar
 */
#if TRIO_WIDECHAR
TRIO_PRIVATE int
TrioReadWideChar
TRIO_ARGS4((self, target, flags, width),
            trio_class_t *self,
            trio_wchar_t *target,
            trio_flags_t flags,
            int width)
{
    int i;
    int j;
    int size;
    int amount = 0;
    trio_wchar_t wch;
    char buffer[MB_LEN_MAX + 1];

    assert(VALID(self));
    assert(VALID(self->InStream));

    for (i = 0;
         (self->current != EOF) && (i < width);
         i++)
    {
        if (isascii(self->current))
        {
            if (TrioReadChar(self, buffer, flags, 1) == 0)

```

```

        return 0;
        buffer[1] = NIL;
    }
    else
    {
        /*
         * Collect a multibyte character, by enlarging buffer until
         * it contains a fully legal multibyte character, or the
         * buffer is full.
         */
        j = 0;
        do
        {
            buffer[j++] = (char)self->current;
            buffer[j] = NIL;
            self->InStream(self, NULL);
        }
        while ((j < (int)sizeof(buffer)) && (mblen(buffer, (size_t)j) != j));
    }
    if (target)
    {
        size = mbtowc(&wch, buffer, sizeof(buffer));
        if (size > 0)
            target[i] = wch;
    }
    amount += size;
    self->InStream(self, NULL);
}
return amount;
}
#endif /* TRIO_WIDECHAR */

/*****
 * TrioReadWideString
 */
#if TRIO_WIDECHAR
TRIO_PRIVATE BOOLEAN_T
TrioReadWideString
TRIO_ARGS4((self, target, flags, width),
    trio_class_t *self,
    trio_wchar_t *target,
    trio_flags_t flags,
    int width)
{
    int i;
    int size;

    assert(VALID(self));
    assert(VALID(self->InStream));

    TrioSkipWhitespaces(self);

    #if defined(TRIO_COMPILER_SUPPORTS_MULTIBYTE)
        (void)mblen(NULL, 0);
    #endif

```

```

#endif

/*
 * Continue until end of string is reached, a whitespace is encountered,
 * or width is exceeded
 */
for (i = 0;
     ((width == NO_WIDTH) || (i < width)) &&
     (!(self->current == EOF) || isspace(self->current)));
{
    size = TrioReadWideChar(self, &target[i], flags, 1);
    if (size == 0)
        break; /* for */

    i += size;
}
if (target)
    target[i] = WCONST('\0');
return TRUE;
}
#endif /* TRIO_WIDECHAR */

/*****
 * TrioReadGroup
 *
 * FIXME: characterclass does not work with multibyte characters
 */
TRIO_PRIVATE BOOLEAN_T
TrioReadGroup
TRIO_ARGS5((self, target, characterclass, flags, width),
            trio_class_t *self,
            char *target,
            int *characterclass,
            trio_flags_t flags,
            int width)
{
    int ch;
    int i;

    assert(VALID(self));
    assert(VALID(self->InStream));

    ch = self->current;
    for (i = 0;
         ((width == NO_WIDTH) || (i < width)) &&
         (!(ch == EOF) ||
          (((flags & FLAGS_EXCLUDE) != 0) ^ (characterclass[ch] == 0))));
        i++)
    {
        if (target)
            target[i] = (char)ch;
        self->InStream(self, &ch);
    }
}

```

```

    if (target)
        target[i] = NIL;
    return TRUE;
}

/*****
 * TrioReadDouble
 *
 * FIXME:
 *  add long double
 *  handle base
 */
TRIO_PRIVATE BOOLEAN_T
TrioReadDouble
TRIO_ARGS4((self, target, flags, width),
    trio_class_t *self,
    trio_pointer_t target,
    trio_flags_t flags,
    int width)
{
    int ch;
    char doubleString[512];
    int index = 0;
    int start;
    int j;
    BOOLEAN_T isHex = FALSE;

    doubleString[0] = 0;

    if ((width == NO_WIDTH) || (width > (int)sizeof(doubleString) - 1))
        width = sizeof(doubleString) - 1;

    TrioSkipWhitespaces(self);

    /*
     * Read entire double number from stream. trio_to_double requires
     * a string as input, but InStream can be anything, so we have to
     * collect all characters.
     */
    ch = self->current;
    if ((ch == '+') || (ch == '-'))
    {
        doubleString[index++] = (char)ch;
        self->InStream(self, &ch);
        width--;
    }

    start = index;
    switch (ch)
    {
        case 'n':
        case 'N':
            /* Not-a-number */
            if (index != 0)

```

```

break;
/* FALLTHROUGH */
case 'i':
case 'I':
/* Infinity */
while (isalpha(ch) && (index - start < width))
{
doubleString[index++] = (char)ch;
self->InStream(self, &ch);
}
doubleString[index] = NIL;

/* Case insensitive string comparison */
if (trio_equal(&doubleString[start], INFINITE_UPPER) ||
trio_equal(&doubleString[start], LONG_INFINITE_UPPER))
{
if (flags & FLAGS_LONGDOUBLE)
{
if ((start == 1) && (doubleString[0] == '-'))
{
*((trio_long_double_t *)target) = trio_ninf();
}
else
{
*((trio_long_double_t *)target) = trio_pinf();
}
}
else
{
if ((start == 1) && (doubleString[0] == '-'))
{
*((double *)target) = trio_ninf();
}
else
{
*((double *)target) = trio_pinf();
}
}
return TRUE;
}

if (trio_equal(doubleString, NAN_UPPER))
{
/* NaN must not have a preceeding + nor - */
if (flags & FLAGS_LONGDOUBLE)
{
*((trio_long_double_t *)target) = trio_nan();
}
else
{
*((double *)target) = trio_nan();
}
return TRUE;
}

return FALSE;

```



```

        case '0':
            doubleString[index++] = (char)ch;
            self->InStream(self, &ch);
            if (trio_to_upper(ch) == 'X')
        {
            isHex = TRUE;
            doubleString[index++] = (char)ch;
            self->InStream(self, &ch);
        }
        break;

        default:
            break;
    }

    while ((ch != EOF) && (index - start < width))
    {
        /* Integer part */
        if (isHex ? isxdigit(ch) : isdigit(ch))
        {
            doubleString[index++] = (char)ch;
            self->InStream(self, &ch);
        }
        else if (flags & FLAGS_QUOTE)
        {
            /* Compare with thousands separator */
            for (j = 0; internalThousandSeparator[j] && self->current; j++)
            {
                if (internalThousandSeparator[j] != self->current)
                    break;
            }

            self->InStream(self, &ch);
        }
        if (internalThousandSeparator[j])
            break; /* Mismatch */
        else
            continue; /* Match */
    }
    else
        break; /* while */
    }
    if (ch == '.')
    {
        /* Decimal part */
        doubleString[index++] = (char)ch;
        self->InStream(self, &ch);
        while ((isHex ? isxdigit(ch) : isdigit(ch)) &&
            (index - start < width))
        {
            doubleString[index++] = (char)ch;
            self->InStream(self, &ch);
        }
        if (isHex ? (trio_to_upper(ch) == 'P') : (trio_to_upper(ch) == 'E'))
    {

```

```

/* Exponent */
doubleString[index++] = (char)ch;
self->InStream(self, &ch);
if ((ch == '+') || (ch == '-'))
{
    doubleString[index++] = (char)ch;
    self->InStream(self, &ch);
}
while (isdigit(ch) && (index - start < width))
{
    doubleString[index++] = (char)ch;
    self->InStream(self, &ch);
}
}
}

if ((index == start) || (*doubleString == NIL))
    return FALSE;

doubleString[index] = 0;

if (flags & FLAGS_LONGDOUBLE)
{
    *((trio_long_double_t *)target) = trio_to_long_double(doubleString, NULL);
}
else
{
    *((double *)target) = trio_to_double(doubleString, NULL);
}
return TRUE;
}

/*****
* TrioReadPointer
*/
TRIO_PRIVATE BOOLEAN_T
TrioReadPointer
TRIO_ARGS3((self, target, flags),
    trio_class_t *self,
    trio_pointer_t *target,
    trio_flags_t flags)
{
    trio_uintmax_t number;
    char buffer[sizeof(internalNullString)];

    flags |= (FLAGS_UNSIGNED | FLAGS_ALTERNATIVE | FLAGS_NILPADDING);

    if (TrioReadNumber(self,
        &number,
        flags,
        POINTER_WIDTH,
        BASE_HEX))
    {
        /*
         * The strange assignment of number is a workaround for a compiler

```

```

        * warning
        */
        if (target)
*target = (char *)0 + number;
        return TRUE;
    }
    else if (TrioReadString(self,
(flags & FLAGS_IGNORE)
? NULL
: buffer,
0,
sizeof(internalNullString) - 1))
    {
        if (trio_equal_case(buffer, internalNullString))
    {
        if (target)
            *target = NULL;
        return TRUE;
    }
    }
    return FALSE;
}

/*****
* TrioScanProcess
*/
TRIO_PRIVATE int
TrioScanProcess
TRIO_ARGS3((data, format, parameters),
    trio_class_t *data,
    TRIO_CONST char *format,
    trio_parameter_t *parameters)
{
#ifdef TRIO_COMPILER_SUPPORTS_MULTIBYTE
    int charlen;
    int cnt;
#endif
    int assignment;
    int ch;
    int index; /* Index of format string */
    int i; /* Index of current parameter */
    trio_flags_t flags;
    int width;
    int base;
    trio_pointer_t pointer;

    assignment = 0;
    i = 0;
    index = 0;
    data->InStream(data, &ch);

#ifdef TRIO_COMPILER_SUPPORTS_MULTIBYTE
    (void)mblen(NULL, 0);
#endif
}

```

```

    while (format[index])
    {
#ifdef TRIO_COMPILER_SUPPORTS_MULTIBYTE
        if (! isascii(format[index]))
        {
            charlen = mblen(&format[index], MB_LEN_MAX);
            if (charlen != -1)
            {
                /* Compare multibyte characters in format string */
                for (cnt = 0; cnt < charlen - 1; cnt++)
                {
                    if (ch != format[index + cnt])
                    {
                        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
                    }
                }
                data->InStream(data, &ch);
            }
            continue; /* while characters left in formatting string */
        }
    }
#endif /* TRIO_COMPILER_SUPPORTS_MULTIBYTE */

    if ((EOF == ch) && (parameters[i].type != FORMAT_COUNT))
    {
        return (assignment > 0) ? assignment : EOF;
    }

    if (CHAR_IDENTIFIER == format[index])
    {
        if (CHAR_IDENTIFIER == format[index + 1])
        {
            /* Two % in format matches one % in input stream */
            if (CHAR_IDENTIFIER == ch)
            {
                data->InStream(data, &ch);
                index += 2;
                continue; /* while format chars left */
            }
        }
        else
        {
            return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
        }
    }

    /* Skip the parameter entries */
    while (parameters[i].type == FORMAT_PARAMETER)
        i++;

    flags = parameters[i].flags;
    /* Find width */
    width = parameters[i].width;
    if (flags & FLAGS_WIDTH_PARAMETER)
    {
        /* Get width from parameter list */
        width = (int)parameters[width].data.number.as_signed;
    }

```

```

/* Find base */
base = parameters[i].base;
if (flags & FLAGS_BASE_PARAMETER)
{
    /* Get base from parameter list */
    base = (int)parameters[base].data.number.as_signed;
}

switch (parameters[i].type)
{
    case FORMAT_INT:
    {
        trio_uintmax_t number;

        if (0 == base)
            base = BASE_DECIMAL;

        if (!TrioReadNumber(data,
            &number,
            flags,
            width,
            base))
            return assignment;

        if (!(flags & FLAGS_IGNORE))
        {
            assignment++;

            pointer = parameters[i].data.pointer;
            #if defined(QUALIFIER_SIZE_T) || defined(QUALIFIER_SIZE_T_UPPER)
            if (flags & FLAGS_SIZE_T)
                *(size_t *)pointer = (size_t)number;
            else
            #endif
            #if defined(QUALIFIER_PTRDIFF_T)
            if (flags & FLAGS_PTRDIFF_T)
                *(ptrdiff_t *)pointer = (ptrdiff_t)number;
            else
            #endif
            #if defined(QUALIFIER_INTMAX_T)
            if (flags & FLAGS_INTMAX_T)
                *(trio_intmax_t *)pointer = (trio_intmax_t)number;
            else
            #endif
            if (flags & FLAGS_QUAD)
                *(trio_ulonglong_t *)pointer = (trio_ulonglong_t)number;
            else if (flags & FLAGS_LONG)
                *(long int *)pointer = (long int)number;
            else if (flags & FLAGS_SHORT)
                *(short int *)pointer = (short int)number;
            else
                *(int *)pointer = (int)number;
        }
    }
}

```

```

        break; /* FORMAT_INT */

        case FORMAT_STRING:
#ifdef TRIO_WIDECHAR
            if (flags & FLAGS_WIDECHAR)
            {
                if (!TrioReadWideString(data,
                    (flags & FLAGS_IGNORE)
                    ? NULL
                    : parameters[i].data.wstring,
                    flags,
                    width))
                    return assignment;
            }
            else
#endif
            {
                if (!TrioReadString(data,
                    (flags & FLAGS_IGNORE)
                    ? NULL
                    : parameters[i].data.string,
                    flags,
                    width))
                    return assignment;
            }
            if (!(flags & FLAGS_IGNORE))
                assignment++;
            break; /* FORMAT_STRING */

        case FORMAT_DOUBLE:
        {
            trio_pointer_t pointer;

            if (flags & FLAGS_IGNORE)
            {
                pointer = NULL;
            }
            else
            {
                pointer = (flags & FLAGS_LONGDOUBLE)
                    ? (trio_pointer_t)parameters[i].data.longdoublePointer
                    : (trio_pointer_t)parameters[i].data.doublePointer;
            }
            if (!TrioReadDouble(data, pointer, flags, width))
            {
                return assignment;
            }
            if (!(flags & FLAGS_IGNORE))
            {
                assignment++;
            }
            break; /* FORMAT_DOUBLE */
        }
        case FORMAT_GROUP:
        {

```

```

int characterclass[MAX_CHARACTER_CLASS + 1];
int rc;

/* Skip over modifiers */
while (format[index] != SPECIFIER_GROUP)
{
    index++;
}
/* Skip over group specifier */
index++;

memset(characterclass, 0, sizeof(characterclass));
rc = TrioGetCharacterClass(format,
    &index,
    &flags,
    characterclass);
if (rc < 0)
    return rc;

if (!TrioReadGroup(data,
    (flags & FLAGS_IGNORE)
    ? NULL
    : parameters[i].data.string,
    characterclass,
    flags,
    parameters[i].width))
    return assignment;
if (!(flags & FLAGS_IGNORE))
    assignment++;
}
break; /* FORMAT_GROUP */

case FORMAT_COUNT:
    pointer = parameters[i].data.pointer;
    if (NULL != pointer)
    {
        int count = data->committed;
        if (ch != EOF)
            count--; /* a character is read, but is not consumed yet */
#ifdef QUALIFIER_SIZE_T || defined(QUALIFIER_SIZE_T_UPPER)
        if (flags & FLAGS_SIZE_T)
            *(size_t *)pointer = (size_t)count;
        else
#endif
#ifdef QUALIFIER_PTRDIFF_T
        if (flags & FLAGS_PTRDIFF_T)
            *(ptrdiff_t *)pointer = (ptrdiff_t)count;
        else
#endif
#ifdef QUALIFIER_INTMAX_T
        if (flags & FLAGS_INTMAX_T)
            *(trio_intmax_t *)pointer = (trio_intmax_t)count;
        else
#endif
    }

```

```

    if (flags & FLAGS_QUAD)
    {
        *(trio_ulonglong_t *)pointer = (trio_ulonglong_t)count;
    }
    else if (flags & FLAGS_LONG)
    {
        *(long int *)pointer = (long int)count;
    }
    else if (flags & FLAGS_SHORT)
    {
        *(short int *)pointer = (short int)count;
    }
    else
    {
        *(int *)pointer = (int)count;
    }
}
break; /* FORMAT_COUNT */

case FORMAT_CHAR:
#ifdef TRIO_WIDECHAR
    if (flags & FLAGS_WIDECHAR)
    {
        if (TrioReadWideChar(data,
            (flags & FLAGS_IGNORE)
            ? NULL
            : parameters[i].data.wstring,
            flags,
            (width == NO_WIDTH) ? 1 : width) == 0)
            return assignment;
    }
    else
#endif
    {
        if (TrioReadChar(data,
            (flags & FLAGS_IGNORE)
            ? NULL
            : parameters[i].data.string,
            flags,
            (width == NO_WIDTH) ? 1 : width) == 0)
            return assignment;
    }
    if (!(flags & FLAGS_IGNORE))
assignment++;
    break; /* FORMAT_CHAR */

case FORMAT_POINTER:
    if (!TrioReadPointer(data,
        (flags & FLAGS_IGNORE)
        ? NULL
        : (trio_pointer_t *)parameters[i].data.pointer,
        flags))
return assignment;
    if (!(flags & FLAGS_IGNORE))
assignment++;

```



```

        break; /* FORMAT_POINTER */

    case FORMAT_PARAMETER:
        break; /* FORMAT_PARAMETER */

    default:
        return TRIO_ERROR_RETURN(TRIO_EINVAL, index);
    }
    ch = data->current;
    index = parameters[i].indexAfterSpecifier;
    i++;
}
    else /* Not an % identifier */
{
    if (isspace((int)format[index]))
    {
        /* Whitespaces may match any amount of whitespaces */
        ch = TrioSkipWhitespaces(data);
    }
    else if (ch == format[index])
    {
        data->InStream(data, &ch);
    }
    else
        return assignment;

    index++;
}
}
    return assignment;
}

/*****
 * TrioScan
 */
TRIO_PRIVATE int
TrioScan
TRIO_ARGS6((source, sourceSize, InStream, format, arglist, argarray),
    trio_pointer_t source,
    size_t sourceSize,
    void (*InStream) TRIO_PROTO((trio_class_t *, int *)),
    TRIO_CONST char *format,
    TRIO_VA_LIST_PTR arglist,
    trio_pointer_t *argarray)
{
    int status;
    trio_parameter_t parameters[MAX_PARAMETERS];
    trio_class_t data;

    assert(VALID(InStream));
    assert(VALID(format));

    memset(&data, 0, sizeof(data));
    data.InStream = InStream;

```

```

    data.location = (trio_pointer_t)source;
    data.max = sourceSize;
    data.error = 0;

#ifdef USE_LOCALE
    if (NULL == internalLocaleValues)
    {
        TrioSetLocale();
    }
#endif

    status = TrioParse(TYPE_SCAN, format, parameters, arglist, argarray);
    if (status < 0)
        return status;

    status = TrioScanProcess(&data, format, parameters);
    if (data.error != 0)
    {
        status = data.error;
    }
    return status;
}

/*****
 * TrioInStreamFile
 */
TRIO_PRIVATE void
TrioInStreamFile
TRIO_ARGS2((self, intPointer),
    trio_class_t *self,
    int *intPointer)
{
    FILE *file;

    assert(VALID(self));
    assert(VALID(self->location));
    assert(VALID(file));

    file = (FILE *)self->location;

    self->current = fgetc(file);
    if (self->current == EOF)
    {
        self->error = (ferror(file))
? TRIO_ERROR_RETURN(TRIO_ERRNO, 0)
: TRIO_ERROR_RETURN(TRIO_EOF, 0);
    }
    else
    {
        self->processed++;
        self->committed++;
    }

    if (VALID(intPointer))
    {

```

```

        *intPointer = self->current;
    }
}

/*****
 * TrioInStreamFileDescriptor
 */
TRIO_PRIVATE void
TrioInStreamFileDescriptor
TRIO_ARGS2((self, intPointer),
    trio_class_t *self,
    int *intPointer)
{
    int fd;
    int size;
    unsigned char input;

    assert(VALID(self));
    assert(VALID(self->location));

    fd = *((int *)self->location);

    size = read(fd, &input, sizeof(char));
    if (size == -1)
    {
        self->error = TRIO_ERROR_RETURN(TRIO_ERRNO, 0);
        self->current = EOF;
    }
    else
    {
        self->current = (size == 0) ? EOF : input;
    }
    if (self->current != EOF)
    {
        self->committed++;
        self->processed++;
    }

    if (VALID(intPointer))
    {
        *intPointer = self->current;
    }
}

/*****
 * TrioInStreamCustom
 */
TRIO_PRIVATE void
TrioInStreamCustom
TRIO_ARGS2((self, intPointer),
    trio_class_t *self,
    int *intPointer)
{
    trio_custom_t *data;

```

```

    assert(VALID(self));
    assert(VALID(self->location));

    data = (trio_custom_t *)self->location;

    self->current = (data->stream.in == NULL)
        ? NIL
        : (data->stream.in)(data->closure);

    if (self->current == NIL)
    {
        self->current = EOF;
    }
    else
    {
        self->processed++;
        self->committed++;
    }

    if (VALID(intPointer))
    {
        *intPointer = self->current;
    }
}

/*****
 * TrioInStreamString
 */
TRIO_PRIVATE void
TrioInStreamString
TRIO_ARGS2((self, intPointer),
    trio_class_t *self,
    int *intPointer)
{
    unsigned char **buffer;

    assert(VALID(self));
    assert(VALID(self->location));

    buffer = (unsigned char **)self->location;
    self->current = (*buffer)[0];
    if (self->current == NIL)
    {
        self->current = EOF;
    }
    else
    {
        (*buffer)++;
        self->processed++;
        self->committed++;
    }

    if (VALID(intPointer))
    {

```

```

        *intPointer = self->current;
    }
}

/*****
 *
 * Formatted scanning functions
 *
 *****/

#if defined(TRIO_DOCUMENTATION)
# include "doc/doc_scanf.h"
#endif
/** @addtogroup Scanf
 * @
 */

/*****
 * scanf
 */

/**
 Scan characters from standard input stream.

 @param format Formatting string.
 @param ... Arguments.
 @return Number of scanned characters.
 */
TRIO_PUBLIC int
trio_scanf
TRIO_VARGS2((format, va_alist),
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioScan((trio_pointer_t)stdin, 0,
        TrioInStreamFile,
        format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vscanf
TRIO_ARGS2((format, args),
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(format));

```

```

    return TrioScan((trio_pointer_t)stdin, 0,
        TrioInStreamFile,
        format, TRIO_VA_LIST_ADDR(args), NULL);
}

TRIO_PUBLIC int
trio_scanfv
TRIO_ARGS2((format, args),
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    assert(VALID(format));

    return TrioScan((trio_pointer_t)stdin, 0,
        TrioInStreamFile,
        format, NULL, args);
}

/*****
 * fscanf
 */
TRIO_PUBLIC int
trio_fscanf
TRIO_VARGS3((file, format, va_alist),
    FILE *file,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(file));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioScan((trio_pointer_t)file, 0,
        TrioInStreamFile,
        format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vfscanf
TRIO_ARGS3((file, format, args),
    FILE *file,
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(file));
    assert(VALID(format));

    return TrioScan((trio_pointer_t)file, 0,
        TrioInStreamFile,

```

```

    format, TRIO_VA_LIST_ADDR(args), NULL);
}

TRIO_PUBLIC int
trio_fscanfv
TRIO_ARGS3((file, format, args),
    FILE *file,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    assert(VALID(file));
    assert(VALID(format));

    return TrioScan((trio_pointer_t)file, 0,
        TrioInStreamFile,
        format, NULL, args);
}

/*****
 * dscanf
 */
TRIO_PUBLIC int
trio_dscanf
TRIO_VARGS3((fd, format, va_alist),
    int fd,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioScan((trio_pointer_t)&fd, 0,
        TrioInStreamFileDescriptor,
        format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vdscanf
TRIO_ARGS3((fd, format, args),
    int fd,
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(format));

    return TrioScan((trio_pointer_t)&fd, 0,
        TrioInStreamFileDescriptor,
        format, TRIO_VA_LIST_ADDR(args), NULL);
}

```

```

TRIO_PUBLIC int
trio_dscanf
TRIO_ARGS3((fd, format, args),
    int fd,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    assert(VALID(format));

    return TrioScan((trio_pointer_t)&fd, 0,
        TrioInStreamFileDescriptor,
        format, NULL, args);
}

/*****
* cscanf
*/
TRIO_PUBLIC int
trio_cscanf
TRIO_VARS4((stream, closure, format, va_alist),
    trio_instream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    data.stream.in = stream;
    data.closure = closure;
    status = TrioScan(&data, 0, TrioInStreamCustom, format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vscanf
TRIO_ARGS4((stream, closure, format, args),
    trio_instream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    va_list args)
{
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));

    data.stream.in = stream;

```



```

    data.closure = closure;
    return TrioScan(&data, 0, TrioInStreamCustom, format, TRIO_VA_LIST_ADDR(args), NULL);
}

TRIO_PUBLIC int
trio_cscanfv
TRIO_ARGS4((stream, closure, format, args),
    trio_instream_t stream,
    trio_pointer_t closure,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    trio_custom_t data;

    assert(VALID(stream));
    assert(VALID(format));

    data.stream.in = stream;
    data.closure = closure;
    return TrioScan(&data, 0, TrioInStreamCustom, format, NULL, args);
}

/*****
 * sscanf
 */
TRIO_PUBLIC int
trio_sscanf
TRIO_VARGS3((buffer, format, va_alist),
    TRIO_CONST char *buffer,
    TRIO_CONST char *format,
    TRIO_VA_DECL)
{
    int status;
    va_list args;

    assert(VALID(buffer));
    assert(VALID(format));

    TRIO_VA_START(args, format);
    status = TrioScan((trio_pointer_t)&buffer, 0,
        TrioInStreamString,
        format, TRIO_VA_LIST_ADDR(args), NULL);
    TRIO_VA_END(args);
    return status;
}

TRIO_PUBLIC int
trio_vsscanf
TRIO_ARGS3((buffer, format, args),
    TRIO_CONST char *buffer,
    TRIO_CONST char *format,
    va_list args)
{
    assert(VALID(buffer));

```

```

    assert(VALID(format));

    return TrioScan((trio_pointer_t)&buffer, 0,
        TrioInStreamString,
        format, TRIO_VA_LIST_ADDR(args), NULL);
}

TRIO_PUBLIC int
trio_sscanfv
TRIO_ARGS3((buffer, format, args),
    TRIO_CONST char *buffer,
    TRIO_CONST char *format,
    trio_pointer_t *args)
{
    assert(VALID(buffer));
    assert(VALID(format));

    return TrioScan((trio_pointer_t)&buffer, 0,
        TrioInStreamString,
        format, NULL, args);
}

/** @} End of Sscanf documentation module */

/*****
 * trio_strerror
 */
TRIO_PUBLIC TRIO_CONST char *
trio_strerror
TRIO_ARGS1((errorcode),
    int errorcode)
{
    /* Textual versions of the error codes */
    switch (TRIO_ERROR_CODE(errorcode))
    {
        case TRIO_EOF:
            return "End of file";
        case TRIO_EINVAL:
            return "Invalid argument";
        case TRIO_ETOOMANY:
            return "Too many arguments";
        case TRIO_EDBLREF:
            return "Double reference";
        case TRIO_EGAP:
            return "Reference gap";
        case TRIO_ENOMEM:
            return "Out of memory";
        case TRIO_ERANGE:
            return "Invalid range";
        case TRIO_ECUSTOM:
            return "Custom error";
        default:
            return "Unknown";
    }
}

```

1.83 linux-windriver

1.83.1 Available under license

NOTE! This copyright does **not** cover user programs that use kernel services by normal system calls - this is merely considered normal use of the kernel, and does **not** fall under the heading of "derived work". Also note that the GPL below is copyrighted by the Free Software Foundation, but the instance of code that it refers to (the Linux kernel) is copyrighted by me and others who actually wrote it.

Also note that the only valid version of the GPL as far as the kernel is concerned is *_this_* particular version of the license (ie v2, not v2.2 or v3.x or whatever), unless explicitly otherwise stated.

Linus Torvalds

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.84 lldpd

1.84.1 Available under license

Files: *

Copyright: © 2008 Vincent Bernat <bernat@luffy.cx>

© 2003, 2004 Henning Brauer <henning@openbsd.org>

© 2006 Pierre-Yves Ritschard <pyr@openbsd.org>

License: MIT

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

On Debian systems, the complete text of the MIT can be found in
`/usr/share/common-licenses/MIT`.

1.85 logrotate

1.85.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.86 Izo

1.86.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

/* lzo_init.c -- initialization of the LZO library

This file is part of the LZO real-time data compression library.

Copyright (C) 1996-2015 Markus Franz Xavier Johannes Oberhumer
All Rights Reserved.

The LZO library is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

The LZO library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with the LZO library; see the file COPYING.

If not, write to the Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Markus F.X.J. Oberhumer
<markus@oberhumer.com>
<http://www.oberhumer.com/opensource/lzo/>
*/

#include "lzo_conf.h"

/*

// Runtime check of the assumptions about the size of builtin types,
// memory model, byte order and other low-level constructs.
//
// We are really paranoid here - LZO should either fail


```

// at startup or not at all.
//
// Because of inlining much of these functions evaluates to nothing.
//
// And while many of the tests seem highly obvious and redundant they are
// here to catch compiler/optimizer bugs. Yes, these do exist.
*****/

#if !defined(__LZO_IN_MINILZO)

#define LZO_WANT_ACC_CHK_CH 1
#undef LZOCHK_ASSERT
#include "lzo_sup.h"

    LZOCHK_ASSERT((LZO_UINT32_C(1) << (int)(8*sizeof(LZO_UINT32_C(1))-1)) > 0)
    LZOCHK_ASSERT_IS_SIGNED_T(lzo_int)
    LZOCHK_ASSERT_IS_UNSIGNED_T(lzo_uint)
    #if !(__LZO_UINTPTR_T_IS_POINTER)
        LZOCHK_ASSERT_IS_UNSIGNED_T(lzo_uintptr_t)
    #endif
    LZOCHK_ASSERT(sizeof(lzo_uintptr_t) >= sizeof(lzo_voidp))
    LZOCHK_ASSERT_IS_UNSIGNED_T(lzo_xint)

    #endif
    #undef LZOCHK_ASSERT

    /*****
    //
    *****/

    union lzo_config_check_union {
        lzo_uint a[2];
        unsigned char b[2*LZO_MAX(8,sizeof(lzo_uint))];
    #if defined(lzo_uint64_t)
        lzo_uint64_t c[2];
    #endif
    };

    #if 0
    #define u2p(ptr,off) ((lzo_voidp) (((lzo_bytew)(lzo_voidp)(ptr)) + (off)))
    #else
    static __lzo_noinline lzo_voidp u2p(lzo_voidp ptr, lzo_uint off)
    {
        return (lzo_voidp) ((lzo_bytew) ptr + off);
    }
    #endif

    LZO_PUBLIC(int)
    _lzo_config_check(void)
    {
        #if (LZO_CC_CLANG && (LZO_CC_CLANG >= 0x030100ul && LZO_CC_CLANG < 0x030300ul))

```

```

# if 0
/* work around a clang 3.1 and clang 3.2 compiler bug; clang 3.3 and 3.4 work */
volatile
# endif
#endif
union lzo_config_check_union u;
lzo_voidp p;
unsigned r = 1;

u.a[0] = u.a[1] = 0;
p = u2p(&u, 0);
r &= ((* (lzo_bytep) p) == 0);
#if !(LZO_CFG_NO_CONFIG_CHECK)
#if (LZO_ABI_BIG_ENDIAN)
u.a[0] = u.a[1] = 0; u.b[sizeof(lzo_uint) - 1] = 128;
p = u2p(&u, 0);
r &= ((* (lzo_uintp) p) == 128);
#endif
#if (LZO_ABI_LITTLE_ENDIAN)
u.a[0] = u.a[1] = 0; u.b[0] = 128;
p = u2p(&u, 0);
r &= ((* (lzo_uintp) p) == 128);
#endif
#endif
u.a[0] = u.a[1] = 0;
u.b[0] = 1; u.b[3] = 2;
p = u2p(&u, 1);
r &= UA_GET_NE16(p) == 0;
r &= UA_GET_LE16(p) == 0;
u.b[1] = 128;
r &= UA_GET_LE16(p) == 128;
u.b[2] = 129;
r &= UA_GET_LE16(p) == LZO_UINT16_C(0x8180);
#if (LZO_ABI_BIG_ENDIAN)
r &= UA_GET_NE16(p) == LZO_UINT16_C(0x8081);
#endif
#if (LZO_ABI_LITTLE_ENDIAN)
r &= UA_GET_NE16(p) == LZO_UINT16_C(0x8180);
#endif
u.a[0] = u.a[1] = 0;
u.b[0] = 3; u.b[5] = 4;
p = u2p(&u, 1);
r &= UA_GET_NE32(p) == 0;
r &= UA_GET_LE32(p) == 0;
u.b[1] = 128;
r &= UA_GET_LE32(p) == 128;
u.b[2] = 129; u.b[3] = 130; u.b[4] = 131;
r &= UA_GET_LE32(p) == LZO_UINT32_C(0x83828180);
#if (LZO_ABI_BIG_ENDIAN)
r &= UA_GET_NE32(p) == LZO_UINT32_C(0x80818283);
#endif
#if (LZO_ABI_LITTLE_ENDIAN)
r &= UA_GET_NE32(p) == LZO_UINT32_C(0x83828180);
#endif
#if defined(UA_GET_NE64)
u.c[0] = u.c[1] = 0;

```

```

    u.b[0] = 5; u.b[9] = 6;
    p = u2p(&u, 1);
    u.c[0] = u.c[1] = 0;
    r &= UA_GET_NE64(p) == 0;
#if defined(UA_GET_LE64)
    r &= UA_GET_LE64(p) == 0;
    u.b[1] = 128;
    r &= UA_GET_LE64(p) == 128;
#endif
#endif
#if defined(lzo_bitops_ctlz32)
    { unsigned i = 0; lzo_uint32_t v;
      for (v = 1; v != 0 && r == 1; v <=<= 1, i++) {
        r &= lzo_bitops_ctlz32(v) == 31 - i;
        r &= lzo_bitops_ctlz32_func(v) == 31 - i;
      }
    }
#endif
#if defined(lzo_bitops_ctlz64)
    { unsigned i = 0; lzo_uint64_t v;
      for (v = 1; v != 0 && r == 1; v <=<= 1, i++) {
        r &= lzo_bitops_ctlz64(v) == 63 - i;
        r &= lzo_bitops_ctlz64_func(v) == 63 - i;
      }
    }
#endif
#if defined(lzo_bitops_cttz32)
    { unsigned i = 0; lzo_uint32_t v;
      for (v = 1; v != 0 && r == 1; v <=<= 1, i++) {
        r &= lzo_bitops_cttz32(v) == i;
        r &= lzo_bitops_cttz32_func(v) == i;
      }
    }
#endif
#if defined(lzo_bitops_cttz64)
    { unsigned i = 0; lzo_uint64_t v;
      for (v = 1; v != 0 && r == 1; v <=<= 1, i++) {
        r &= lzo_bitops_cttz64(v) == i;
        r &= lzo_bitops_cttz64_func(v) == i;
      }
    }
#endif
#endif
    LZO_UNUSED_FUNC(lzo_bitops_unused_funcs);

    return r == 1 ? LZO_E_OK : LZO_E_ERROR;
}

/*****
//
*****/

LZO_PUBLIC(int)
__lzo_init_v2(unsigned v, int s1, int s2, int s3, int s4, int s5,
              int s6, int s7, int s8, int s9)
{
    int r;

```

```
#if defined(__LZO_IN_MINILZO)
#elif (LZO_CC_MSC && ((_MSC_VER) < 700))
#else
#define LZO_WANT_ACC_CHK_CH 1
#undef LZOCHK_ASSERT
#define LZOCHK_ASSERT(expr) LZO_COMPILE_TIME_ASSERT(expr)
#include "lzo_suppl.h"
#endif
#undef LZOCHK_ASSERT

if (v == 0)
    return LZO_E_ERROR;

r = (s1 == -1 || s1 == (int) sizeof(short)) &&
    (s2 == -1 || s2 == (int) sizeof(int)) &&
    (s3 == -1 || s3 == (int) sizeof(long)) &&
    (s4 == -1 || s4 == (int) sizeof(lzo_uint32_t)) &&
    (s5 == -1 || s5 == (int) sizeof(lzo_uint)) &&
    (s6 == -1 || s6 == (int) lzo_sizeof_dict_t) &&
    (s7 == -1 || s7 == (int) sizeof(char *)) &&
    (s8 == -1 || s8 == (int) sizeof(lzo_voidp)) &&
    (s9 == -1 || s9 == (int) sizeof(lzo_callback_t));
if (!r)
    return LZO_E_ERROR;

r = _lzo_config_check();
if (r != LZO_E_OK)
    return r;

return r;
}

#if !defined(__LZO_IN_MINILZO)
#include "lzo_dll.ch"
#endif

/* vim:set ts=4 sw=4 et: */
```

1.87 m4

1.87.1 Available under license

The files in this directory provide example uses of GNU M4.
The following copyright notice applies to each of these
description files.

Copyright (C) 2006, 2010-2012 Free Software Foundation, Inc.
This file is free software; the Free Software Foundation
gives unlimited permission to copy and/or distribute it,
with or without modifications, as long as this notice is preserved.
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a

computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding

Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product

(including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from

a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work

conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different

permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or

(at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.88 MAKEDEV

1.88.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder say-

ing it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geograph-

ical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w`. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c`
for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.89 mingetty

1.89.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so

that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for

making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under

any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.90 minicom

1.90.1 Available under license

Minicom is Copyright (C) 1991,1992,1993,1994,1995,1996
Miquel van Smoorenburg.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we

want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any

part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED

OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it

under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
/*
 * minicom.hConstants, defaults, globals etc.
 *
 * $Id: minicom.h,v 1.26 2009-06-06 21:19:36 al-guest Exp $
 *
 * This file is part of the minicom communications package,
 * Copyright 1991-1995 Miquel van Smoorenburg.
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version
 * 2 of the License, or (at your option) any later version.
 *
 * You should have received a copy of the GNU General Public License along
 * with this program; if not, write to the Free Software Foundation, Inc.,
 * 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
 *
 * fmg 1/11/94 colors
 * fmg 8/22/97 History Buffer Search support
 * jl 23.06.97 sec parameter to m_dtrtoggle
 * jl 04.09.97 conversion tables
 * jl 09.09.97 loadconv and saveconv protos
 * jl 05.10.97 changed return value of dial() to long
 * jseymour@jimsun.LinxNet.com (Jim Seymour) 03/26/98 - Added prototype
 * for new "get_port()" function in util.c.
 */

/* First include all other application-dependant include files. */
#include "config.h"
#include "configsym.h"
#include "window.h"
#include "keyboard.h"
#include "vt100.h"
```

```
#include "libport.h"

#include <time.h>
#include <stdbool.h>

#if HAVE_LOCKDEV
#include <ttylock.h>
#endif

#ifdef USE_SOCKET
#include <sys/socket.h>
#include <sys/un.h>
#endif

/*
 * kubota@debian.or.jp 08/08/98
 * COLS must be equal to or less than MAXCOLS.
 */
#define MAXCOLS 256

#define XA_OK_EXIST1
#define XA_OK_NOTEXIST2

#ifdef EXTERN
# define EXTERN extern
#endif

#ifdef _UWIN2P0
EXTERN int LINES;
EXTERN int COLS;
#endif

EXTERN int dosetup;

EXTERN char stdattr; /* Standard attribute */

EXTERN WIN *us; /* User screen */
EXTERN WIN *st; /* Status Line */

EXTERN short terminal; /* terminal type */
EXTERN time_t online; /* Time online in minutes */
EXTERN long linespd; /* Line speed */
EXTERN short portfd; /* File descriptor of the serial port. */
EXTERN short lines; /* Nr. of lines on the screen */
EXTERN short cols; /* Nr. of cols of the screen */
EXTERN int keypadmode; /* Mode of keypad */
EXTERN int cursormode; /* Mode of cursor (arrow) keys */

EXTERN int docap; /* Capture data to capture file */
EXTERN FILE *capfp; /* File to capture to */
EXTERN int addlf; /* Add LF after CR */
EXTERN int addcr; /* Insert CR before LF */
EXTERN int wrapln; /* Linewrap default */
EXTERN int display_hex; /* Display in hex */
```

```

EXTERN int tempst; /* Status line is temporary */
EXTERN int escape; /* Escape code. */
EXTERN int option_T_used; /* option -T has been supplied, obsolete, print warning */

EXTERN char lockfile[128]; /* UUCP lock file of terminal */
EXTERN char homedir[256]; /* Home directory of user */
EXTERN char logfile[PARS_VAL_LEN]; /* Name of the logfile */
EXTERN char username[16]; /* Who is using minicom? */

EXTERN int bogus_dcd; /* This indicates the dcd status if no 'real' dcd */
EXTERN int alt_override; /* -m option */

EXTERN char parfile[256]; /* Global parameter file */
EXTERN char pparfile[256]; /* Personal parameter file */

EXTERN char scr_name[256]; /* Name of last script */
EXTERN char scr_user[33]; /* Login name to use with script */
EXTERN char scr_passwd[33]; /* Password to use with script */

EXTERN char termtype[32]; /* Value of getenv("TERM"); */
EXTERN char *dial_tty; /* tty to use. */

EXTERN char *dial_name; /* System we're connected to */
EXTERN char *dial_number; /* Number we've dialed. */
EXTERN char *dial_user; /* Our username there */
EXTERN char *dial_pass; /* Our password */

#ifdef USE_SOCKET
EXTERN int portfd_is_socket; /* File descriptor is a unix socket */
EXTERN int portfd_is_connected; /* 1 if the socket is connected */
EXTERN struct sockaddr_un portfd_sock_addr; /* the unix socket address */
#define portfd_connected ((portfd_is_socket && !portfd_is_connected) \
    ? -1 : portfd)
#else
#define portfd_connected portfd
#define portfd_is_socket 0
#define portfd_is_connected 0
#endif /* USE_SOCKET */

/*
 * fmg 8/22/97
 * Search pattern can be THIS long (x characters)
 */
#define MAX_SEARCH 30

/* fmg 1/11/94 colors */

EXTERN int mfcolor; /* Menu Foreground Color */
EXTERN int mbcolor; /* Menu Background Color */
EXTERN int tfcolor; /* Terminal Foreground Color */
EXTERN int tbcolor; /* Terminal Background Color */
EXTERN int sfcolor; /* Status Bar Foreground Color */
EXTERN int sbcolor; /* Status Bar Background Color */
EXTERN int st_attr; /* Status Bar attributes. */

```

```
/* jl 04.09.97 conversion tables */
EXTERN unsigned char vt_outmap[256], vt_inmap[256];

/* MARK updated 02/17/95 - history buffer */
EXTERN int num_hist_lines; /* History buffer size */

/* fmg 1/11/94 colors - convert color word to # */

int Jcolor(char *);

EXTERN int size_changed; /* Window size has changed */
extern const char *Version; /* Minicom version */

EXTERN int local_echo; /* Local echo on/off. */

/* Forward declaration. */
struct dialent;

/* Global functions */

/* Prototypes from file: config.c */
void read_parms(void);
int waccess(char *s);
int config(int setup);
void get_bbp(char *ba, char *bi, char *pa, char *stopb, int curr_ok);
int update_bbp_from_char(char c, char *ba, char *bi, char *pa, char *stopb,
                        int curr_ok);
const char *yesno(int k);
int dotermmenu(void);
int dodflsave(void); /* fmg - need it */
void vdodflsave(void); /* fmg - need it */
int domacsave(void); /* fmg - need it */
int loadconv(char *); /* jl */
int saveconv(char *); /* jl */
int speed_valid(unsigned int);

/* Prototypes from file: common.c */
char *pfix_home(char *s);
void do_log(const char *line, ...);
size_t one_mbtowc (wchar_t *pwc, const char *s, size_t n);
size_t one_wctomb (char *s, wchar_t wchar);
size_t mbslen (const char *s);

/* Prototypes from file: dial.c */
#ifdef VC_MUSIC
void music(void);
#endif
void mputs(const char *s, int how);
void modeminit(void);
void modemreset(void);
void hangup(void);
void sendbreak(void);
long dial(struct dialent *d, struct dialent **d2);
int readdialdir(void);
```

```

void dialone(char *entry);
void dialdir(void);

/* Prototypes from file: file.c */
char *filedir(int how_many, int downloading);
void init_dir(char dir);

/* Prototypes from file: util.c */
int fastexec(char *cmd);
int fastsystem(char *cmd, char *in, char *out, char *err);
char *get_port(char *);

/* Prototypes from file: help.c */
int help(void);

/* Prototypes from file: ipc.c */
int check_io(int fd1, int fd2, int tmout, char *buf, int buf_size, int *bytes_red);
int keyboard(int cmd, int arg);

/* Prototypes from file: keyserv.c */
void handler(int dummy);
void sendstr(char *s);

/* Prototypes from file: main.c */
extern time_t old_online;
void leave(const char *s) __attribute__((noreturn));
char *esc_key(void);
void term_socket_connect(void);
void term_socket_close(void);
int open_term(int doinit, int show_win_on_error, int no_msgs);
void init_emul(int type, int do_init);
void timer_update(void);
void show_status(void);
void set_status_line_format(const char *s);
void scriptname(const char *s);
int do_terminal(void);
void status_set_display(const char *text, int duration_s);

/* Prototypes from file: minicom.c */
void port_init(void);
void toggle_addlf(void);
void toggle_local_echo(void);

void drawhist_look(WIN *w, int y, int r, wchar_t *look, int case_matters);
void searchhist(WIN *w_hist, wchar_t *str);
int find_next(WIN *w, WIN *w_hist, int hit_line, wchar_t *look,
              int case_matters);
const wchar_t *upcase(wchar_t *dest, wchar_t *src);
wchar_t *StrStr(wchar_t *str1, wchar_t *str2, int case_matters);

void do_iconv(char **inbuf, size_t *inbytesleft,
              char **outbuf, size_t *outbytesleft);
int using_iconv(void);

/* Prototypes from file: rwconf.c */

```



```
int writepars(FILE *fp, int all);
int writemacs(FILE *fp);
int readpars(FILE *fp, enum config_type conftype);
int readmacs(FILE *fp, int init); /* fmg */

/* Prototypes from file: sysdep1.c */
void m_sethwf(int fd, int on);
void m_dtrtoggle(int fd, int sec);
void m_break(int fd);
int m_getdcd(int fd);
void m_setdcd(int fd, int what);
void m_savestate(int fd);
void m_restorestate(int fd);
void m_nohang(int fd);
void m_hupcl(int fd, int on);
void m_flush(int fd);
void m_flush_script(int fd);
unsigned m_getmaxspd(void);
void m_setparms(int fd, char *baudr, char *par, char *bits,
                char *stopb, int hwf, int swf);
int m_wait(int *st);

/* Prototypes from file: sysdep2.c */
void getrowcols(int *rows, int *cols);
int setcbreak(int mode);
void enab_sig(int onoff, int intrchar);

/* Prototypes from file: updown.c */
void updown(int what, int nr);
int mc_setenv(const char *, const char *);
void kermi(void);
void runscript(int ask, const char *s, const char *l, const char *p);
int paste_file(void);

/* Prototypes from file: windiv.c */
WIN *mc_tell(const char *, ...);
void werror(const char *, ...);
int ask(const char *what, const char *s[]);
char *input(char *s, char *buf);

/* Prototypes from file: wildmat.c */
int wildmat(const char *, const char *);

/* Prototypes from file: wkeys.c */
extern int io_pending, pendingkeys;

/* Prototypes from file: config.c */
void domacros(void);

int lockfile_create(int no_msgs);
void lockfile_remove(void);
```

```
/* We want the ANSI offsetof macro to do some dirty stuff. */
#ifndef offsetof
# define offsetof(type, member) ((int) &((type *)0)->member)
#endif

/* Values for the "flags". */
#define FL_ECHO0x01/* Local echo on/off. */
#define FL_DEL0x02/* Backspace or DEL */
#define FL_WRAP0x04/* Use autowrap. */
#define FL_ANSI0x08/* Type of term emulation */
#define FL_TAG0x80/* This entry is tagged. */
#define FL_SAVE0x0f/* Which portions of flags to save. */

enum {
    TIMESTAMP_LINE_OFF,
    TIMESTAMP_LINE_SIMPLE,
    TIMESTAMP_LINE_EXTENDED,
    TIMESTAMP_LINE_PER_SECOND,
    TIMESTAMP_LINE_NR_OF_OPTIONS, // must be last
};
```

1.91 mktemp

1.91.1 Available under license

Mktemp is distributed under the following ISC-style license:

Copyright (c) 1996-1997, 2000-2001, 2008, 2010
Todd C. Miller <Todd.Miller@courtesan.com>
Copyright (c) 1996, David Mazieres <dm@uun.org>
Copyright (c) 2008, Damien Miller <djm@openbsd.org>

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

1.92 mtd-utils

1.92.1 Available under license

```

/*
 * Copyright (c) Artem Bityutskiy, 2007, 2008
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See
 * the GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

#ifndef __MTD_UTILS_COMMON_H__
#define __MTD_UTILS_COMMON_H__

#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <fcntl.h>
#include <errno.h>
#include <features.h>
#include <inttypes.h>
#include "version.h"

#ifndef PROGRAM_NAME
# error "You must define PROGRAM_NAME before including this header"
#endif

#ifdef __cplusplus
extern "C" {
#endif

#ifndef MIN /* some C lib headers define this for us */
#define MIN(a, b) ((a) < (b) ? (a) : (b))
#endif
#ifndef MAX
#define MAX(a, b) ((a) > (b) ? (a) : (b))
#endif
#define min(a, b) MIN(a, b) /* glue for linux kernel source */
#define ARRAY_SIZE(a) (sizeof(a) / sizeof((a)[0]))

```

```

#define ALIGN(x,a) __ALIGN_MASK(x,(typeof(x))(a)-1)
#define __ALIGN_MASK(x,mask) (((x)+(mask))&~(mask))

#define min_t(t,x,y) ({ \
    typeof((x)) _x = (x); \
    typeof((y)) _y = (y); \
    (_x < _y) ? _x : _y; \
})

#define max_t(t,x,y) ({ \
    typeof((x)) _x = (x); \
    typeof((y)) _y = (y); \
    (_x > _y) ? _x : _y; \
})

#ifndef O_CLOEXEC
#define O_CLOEXEC 0
#endif

/* define a print format specifier for off_t */
#ifdef __USE_FILE_OFFSET64
#define PRIxoff_t PRIx64
#define PRIdoff_t PRId64
#else
#define PRIxoff_t "l"PRIx32
#define PRIdoff_t "l"PRId32
#endif

/* Verbose messages */
#define bareverbose(verbose, fmt, ...) do { \
    if (verbose) \
        printf(fmt, ##__VA_ARGS__); \
    } while(0)
#define verbose(verbose, fmt, ...) \
    bareverbose(verbose, "%s: " fmt "\n", PROGRAM_NAME, ##__VA_ARGS__)

/* Normal messages */
#define normmsg_cont(fmt, ...) do { \
    printf("%s: " fmt, PROGRAM_NAME, ##__VA_ARGS__); \
    } while(0)
#define normmsg(fmt, ...) do { \
    normmsg_cont(fmt "\n", ##__VA_ARGS__); \
    } while(0)

/* Error messages */
#define errmsg(fmt, ...) ({ \
    fprintf(stderr, "%s: error!: " fmt "\n", PROGRAM_NAME, ##__VA_ARGS__); \
    -1; \
    })
#define errmsg_die(fmt, ...) do { \
    exit(errmsg(fmt, ##__VA_ARGS__)); \
    } while(0)

```

```

/* System error messages */
#define sys_errmsg(fmt, ...) ({
    int _err = errno;
    errmsg(fmt, ##__VA_ARGS__);
    fprintf(stderr, "%*serror %d (%s)\n", (int)sizeof(PROGRAM_NAME) + 1, \
        "", _err, strerror(_err));
    -1;
})
#define sys_errmsg_die(fmt, ...) do {
    exit(sys_errmsg(fmt, ##__VA_ARGS__));
} while(0)

/* Warnings */
#define warnmsg(fmt, ...) do {
    fprintf(stderr, "%s: warning!: " fmt "\n", PROGRAM_NAME, ##__VA_ARGS__);
} while(0)

/* uClibc versions before 0.9.34 and musl don't have rpmatch() */
#if defined(__UCLIBC__) && \
    (__UCLIBC_MAJOR__ == 0 && \
    (__UCLIBC_MINOR__ < 9 || \
    (__UCLIBC_MINOR__ == 9 && __UCLIBC_SUBLEVEL__ < 34))) || \
    !defined(__GLIBC__)
#undef rpmatch
#define rpmatch __rpmatch
static inline int __rpmatch(const char *resp)
{
    return (resp[0] == 'y' || resp[0] == 'Y') ? 1 :
    (resp[0] == 'n' || resp[0] == 'N') ? 0 : -1;
}
#endif

/**
 * prompt the user for confirmation
 */
static inline bool prompt(const char *msg, bool def)
{
    char *line = NULL;
    size_t len;
    bool ret = def;

    do {
        normsg_cont("%s (%c/%c) ", msg, def ? 'Y' : 'y', def ? 'n' : 'N');
        fflush(stdout);

        while (getline(&line, &len, stdin) == -1) {
            printf("failed to read prompt; assuming '%s'\n",
                def ? "yes" : "no");
            break;
        }

        if (strcmp("\n", line) != 0) {
            switch (rpmatch(line)) {
                case 0: ret = false; break;
                case 1: ret = true; break;
            }
        }
    } while (0);
}

```

```

case -1:
puts("unknown response; please try again");
continue;
}
}
break;
} while (1);

free(line);

return ret;
}

static inline int is_power_of_2(unsigned long long n)
{
return (n != 0 && ((n & (n - 1)) == 0));
}

/**
 * simple_strtoX - convert a hex/dec/oct string into a number
 * @snum: buffer to convert
 * @error: set to 1 when buffer isn't fully consumed
 *
 * These functions are similar to the standard strtX() functions, but they are
 * a little bit easier to use if you want to convert full string of digits into
 * the binary form. The typical usage:
 *
 * int error = 0;
 * unsigned long num;
 *
 * num = simple_strtoul(str, &error);
 * if (error || ... if needed, your check that num is not out of range ...)
 * error_happened();
 */
#define simple_strtoX(func, type) \
static inline type simple_##func(const char *snum, int *error) \
{ \
char *endptr; \
type ret = func(snum, &endptr, 0); \
\
if (error && (!*snum || *endptr)) { \
errmsg("%s: unable to parse the number '%s'", #func, snum); \
*error = 1; \
} \
\
return ret; \
}
simple_strtoX(strtol, long int)
simple_strtoX strtoll, long long int)
simple_strtoX strtoul, unsigned long int)
simple_strtoX strtoull, unsigned long long int)

/* Simple version-printing for utils */
#define common_print_version() \

```

```
do { \
printf("%s %s\n", PROGRAM_NAME, VERSION); \
} while (0)
```

```
#include "xalloc.h"
```

```
#ifdef __cplusplus
}
#endif
```

```
#endif /* !__MTD_UTILS_COMMON_H__ */
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
```

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free

software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in

whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to

refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES

PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author

Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.

1.93 ncurses

1.93.1 Available under license

Copyright 2018-2019,2020 Thomas E. Dickey
Copyright 1998-2017,2018 Free Software Foundation, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, distribute with modifications, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE ABOVE COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name(s) of the above copyright holders shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization.

-- vile:txtmode fc=72
-- \$Id: COPYING,v 1.9 2020/02/08 13:34:12 tom Exp \$

1.94 neon

1.94.1 Available under license

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some
specially designated Free Software Foundation software, and to any
other libraries whose authors decide to use it. You can use it for
your libraries, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid
anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if
you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you. You must make sure that they, too, receive or can get the source
code. If you link a program with the library, you must provide
complete object files to the recipients so that they can relink them
with the library, after making changes to the library and recompiling
it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright
the library, and (2) offer you this license which gives you legal
permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain

that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of

the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by

this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library

facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed

through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR

CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

/*

HTTP utility functions

Copyright (C) 1999-2006, Joe Orton <joe@manyfish.co.uk>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

*/

```
#ifndef NE_UTILS_H
#define NE_UTILS_H
```

```
#include <sys/types.h>
```

```
#include <stdarg.h>
#include <stdio.h>
```

```
#include "ne_defs.h"
```

```
#ifdef NEON_TRIO
#include <trio.h>
#endif
```

```
NE_BEGIN_DECLS
```

```
/* Returns a human-readable library version string describing the
 * version and build information; for example:
 * "neon 0.2.0: Library build, OpenSSL support" */
const char *ne_version_string(void);
```

```
/* Returns non-zero if library version is not of major version
 * 'major', or if minor version is not greater than or equal to
 * 'minor'. For neon versions with major == 0, all minor versions are
 * presumed to be incompatible. */
int ne_version_match(int major, int minor);
```

```
/* Feature codes: */
#define NE_FEATURE_SSL (1) /* SSL/TLS support */
#define NE_FEATURE_ZLIB (2) /* zlib compression in compress interface */
#define NE_FEATURE_IPV6 (3) /* IPv6 is supported in resolver */
#define NE_FEATURE_LFS (4) /* large file support */
#define NE_FEATURE_SOCKS (5) /* SOCKSv5 support */
#define NE_FEATURE_TS_SSL (6) /* Thread-safe SSL/TLS support */
#define NE_FEATURE_I18N (7) /* i18n error message support */
```

```

#define NE_FEATURE_SSPI (8) /* NTLM/Negotiate authentication protocol via SSPI */

/* Returns non-zero if library is built with support for the given
 * NE_FEATURE_ * feature code 'code'. */
int ne_has_support(int feature);

/* Debugging macro to allow code to be optimized out if debugging is
 * disabled at build time. */
#ifndef NE_DEBUGGING
#define NE_DEBUG if (0) ne_debug
#else /* DEBUGGING */
#define NE_DEBUG ne_debug
#endif /* DEBUGGING */

/* Debugging masks. */
#define NE_DBG_SOCKET (1<<0) /* raw socket */
#define NE_DBG_HTTP (1<<1) /* HTTP request/response handling */
#define NE_DBG_XML (1<<2) /* XML parser */
#define NE_DBG_HTTPAUTH (1<<3) /* HTTP authentication (hiding credentials) */
#define NE_DBG_HTTPPLAIN (1<<4) /* plaintext HTTP authentication */
#define NE_DBG_LOCKS (1<<5) /* WebDAV locking */
#define NE_DBG_XMLPARSE (1<<6) /* low-level XML parser */
#define NE_DBG_HTTPBODY (1<<7) /* HTTP response body blocks */
#define NE_DBG_SSL (1<<8) /* SSL/TLS */
#define NE_DBG_FLUSH (1<<30) /* always flush debugging */

/* Send debugging output to 'stream', for all of the given debug
 * channels. To disable debugging, pass 'stream' as NULL and 'mask'
 * as 0. */
void ne_debug_init(FILE *stream, int mask);

/* The current debug mask and stream set by the last call to
 * ne_debug_init. */
extern int ne_debug_mask;
extern FILE *ne_debug_stream;

/* Produce debug output if any of channels 'ch' is enabled for
 * debugging. */
void ne_debug(int ch, const char *, ...) ne_attribute((format(printf, 2, 3)));

/* Storing an HTTP status result */
typedef struct {
    int major_version;
    int minor_version;
    int code; /* Status-Code value */
    int klass; /* Class of Status-Code (1-5) */
    char *reason_phrase;
} ne_status;

/* NB: couldn't use 'class' in ne_status because it would clash with
 * the C++ reserved word. */

/* Parse 'status_line' using the the RFC2616 Status-Line grammar.
 * s->reason_phrase is malloc-allocated if non-NULL, and must be
 * free'd by the caller. Returns 0 on success, in which case all

```

```
* fields of '*s' will be set; or -1 on parse error, in which case
* '*s' is unmodified. */
int ne_parse_statusline(const char *status_line, ne_status *s);

NE_END_DECLS

#endif /* NE_UTILS_H */
```

1.95 net-snmp

1.95.1 Available under license

Various copyrights apply to this package, listed in various separate parts below. Please make sure that you read all the parts.

---- Part 1: CMU/UCD copyright notice: (BSD like) ----

Copyright 1989, 1991, 1992 by Carnegie Mellon University

Derivative Work - 1996, 1998-2000
Copyright 1996, 1998-2000 The Regents of the University of California

All Rights Reserved

Permission to use, copy, modify and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of CMU and The Regents of the University of California not be used in advertising or publicity pertaining to distribution of the software without specific written permission.

CMU AND THE REGENTS OF THE UNIVERSITY OF CALIFORNIA DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL CMU OR THE REGENTS OF THE UNIVERSITY OF CALIFORNIA BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM THE LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

---- Part 2: Networks Associates Technology, Inc copyright notice (BSD) ----

Copyright (c) 2001-2003, Networks Associates Technology, Inc
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the Networks Associates Technology, Inc nor the

names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 3: Cambridge Broadband Ltd. copyright notice (BSD) ----

Portions of this code are copyright (c) 2001-2003, Cambridge Broadband Ltd.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * The name of Cambridge Broadband Ltd. may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 4: Sun Microsystems, Inc. copyright notice (BSD) ----

Copyright (c) 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Use is subject to license terms below.

This distribution may include materials developed by third parties.

Sun, Sun Microsystems, the Sun logo and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the Sun Microsystems, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

----- Part 5: Sparta, Inc copyright notice (BSD) -----

Copyright (c) 2003-2013, Sparta, Inc
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Sparta, Inc nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR

PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 6: Cisco/BUPTNIC copyright notice (BSD) ----

Copyright (c) 2004, Cisco, Inc and Information Network
Center of Beijing University of Posts and Telecommunications.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Cisco, Inc, Beijing University of Posts and Telecommunications, nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 7: Fabasoft R&D Software GmbH & Co KG copyright notice (BSD) ----

Copyright (c) Fabasoft R&D Software GmbH & Co KG, 2003
oss@fabasoft.com
Author: Bernhard Penz <bernhard.penz@fabasoft.com>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * The name of Fabasoft R&D Software GmbH & Co KG or any of its subsidiaries, brand or product names may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

----- Part 8: Apple Inc. copyright notice (BSD) -----

Copyright (c) 2007 Apple Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of Apple Inc. ("Apple") nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY APPLE AND ITS CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL APPLE OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

----- Part 9: ScienceLogic, LLC copyright notice (BSD) -----

Copyright (c) 2009, ScienceLogic, LLC
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of ScienceLogic, LLC nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 10: Lennart Poettering copyright notice (BSD-like) ----

Copyright 2010 Lennart Poettering

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

---- Part 11: IETF copyright notice (BSD) ----

Copyright (c) 2013 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Internet Society, IETF or IETF Trust, nor the names of specific contributors, may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 12: Arista Networks copyright notice (BSD) ----

Copyright (c) 2013, Arista Networks, Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Arista Networks, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR

A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 13: VMware, Inc. copyright notice (BSD) ----

Copyright (c) 2016, VMware, Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of VMware, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 14: USC/Information Sciences Institute copyright notice (BSD) ----

Copyright (c) 2017-2018, Information Sciences Institute
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the

documentation and/or other materials provided with the distribution.

- * Neither the name of Information Sciences Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.96 net-tools

1.96.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

```
This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

```
/*
 * ifconfig This file contains an implementation of the command
 *          that either displays or sets the characteristics of
 *          one or more of the system's networking interfaces.
 *
 * Version:  $Id: ifconfig.c,v 1.58 2008/10/02 23:31:04 ecki Exp $
 *
 * Author:    Fred N. van Kempen, <waltje@uwalnt.nl.mugnet.org>
 *          and others. Copyright 1993 MicroWalt Corporation
 *
 *          This program is free software; you can redistribute it
 *          and/or modify it under the terms of the GNU General
 *          Public License as published by the Free Software
 *          Foundation; either version 2 of the License, or (at
 *          your option) any later version.
 *
 * Patched to support 'add' and 'del' keywords for INET(4) addresses
 * by Mrs. Brisby <mrs.brisby@nimh.org>
 *
 * {1.34} - 19980630 - Arnaldo Carvalho de Melo <acme@conectiva.com.br>
 *          - gettext instead of catgets for i18n
 *          10/1998 - Andi Kleen. Use interface list primitives.
 *          20001008 - Bernd Eckenfels, Patch from RH for setting mtu
 * (default AF was wrong)
 *          20010404 - Arnaldo Carvalho de Melo, use setlocale
 */

#define DFLT_AF "inet"

#include "config.h"

#include <features.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
```

```

#include <netinet/in.h>
#include <net/if.h>
#include <net/if_arp.h>
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netdb.h>

/* Ugh. But libc5 doesn't provide POSIX types. */
#include <asm/types.h>

#ifdef HAVE_HWSLIP
#include <linux/if_slip.h>
#endif

#ifdef HAVE_AF_INET6

#ifndef _LINUX_IN6_H
/*
 * This is in linux/include/net/ipv6.h.
 */

struct in6_ifreq {
    struct in6_addr ifr6_addr;
    __u32 ifr6_prefixlen;
    unsigned int ifr6_ifindex;
};

#endif

#endif /* HAVE_AF_INET6 */

#ifdef HAVE_AF_IPX
#if (__GLIBC__ > 2) || (__GLIBC__ == 2 && __GLIBC_MINOR__ >= 1)
#include <netipx/ipx.h>
#else
#include "ipx.h"
#endif
#endif
#include "net-support.h"
#include "pathnames.h"
#include "version.h"
#include "../intl.h"
#include "interface.h"
#include "sockets.h"
#include "util.h"

char *Release = RELEASE, *Version = "ifconfig 1.42 (2001-04-13)";

int opt_a = 0; /* show all interfaces */

```

```

int opt_v = 0; /* debugging output flag */

int addr_family = 0; /* currently selected AF */

/* for ipv4 add/del modes */
static int get_nmbc_parent(char *parent, unsigned long *nm,
    unsigned long *bc);
static int set_ifstate(char *parent, unsigned long ip,
    unsigned long nm, unsigned long bc,
    int flag);

static int if_print(char *ifname)
{
    int res;

    if (ife_short)
printf(_("Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg\n"));

    if (!ifname) {
res = for_all_interfaces(do_if_print, &opt_a);
    } else {
struct interface *ife;

ife = lookup_interface(ifname);
if (!ife) {
return -1;
}
res = do_if_fetch(ife);
if (res >= 0)
    ife_print(ife);
    }
return res;
}

/* Set a certain interface flag. */
static int set_flag(char *ifname, short flag)
{
    struct ifreq ifr;

    safe_strncpy(ifr.ifr_name, ifname, IFNAMSIZ);
    if (ioctl(skfd, SIOCGIFFLAGS, &ifr) < 0) {
fprintf(stderr, _("%s: ERROR while getting interface flags: %s\n"),
ifname, strerror(errno));
return (-1);
    }
    safe_strncpy(ifr.ifr_name, ifname, IFNAMSIZ);
    ifr.ifr_flags |= flag;
    if (ioctl(skfd, SIOCSIFFLAGS, &ifr) < 0) {
perror("SIOCSIFFLAGS");
return -1;
    }
    return (0);
}

```

```

/* Clear a certain interface flag. */
static int clr_flag(char *ifname, short flag)
{
    struct ifreq ifr;
    int fd;

    if (strchr(ifname, '.')) {
        /* This is a v4 alias interface. Downing it via a socket for
        another AF may have bad consequences. */
        fd = get_socket_for_af(AF_INET);
        if (fd < 0) {
            fprintf(stderr, _("No support for INET on this system.\n"));
            return -1;
        }
    } else
        fd = skfd;

    safe_strncpy(ifr.ifr_name, ifname, IFNAMSIZ);
    if (ioctl(fd, SIOCGIFFLAGS, &ifr) < 0) {
        fprintf(stderr, _("%s: ERROR while getting interface flags: %s\n"),
            ifname, strerror(errno));
        return -1;
    }
    safe_strncpy(ifr.ifr_name, ifname, IFNAMSIZ);
    ifr.ifr_flags &= ~flag;
    if (ioctl(fd, SIOCSIFFLAGS, &ifr) < 0) {
        perror("SIOCSIFFLAGS");
        return -1;
    }
    return (0);
}

/** test is a specified flag is set */
static int test_flag(char *ifname, short flags)
{
    struct ifreq ifr;
    int fd;

    if (strchr(ifname, '.')) {
        /* This is a v4 alias interface. Downing it via a socket for
        another AF may have bad consequences. */
        fd = get_socket_for_af(AF_INET);
        if (fd < 0) {
            fprintf(stderr, _("No support for INET on this system.\n"));
            return -1;
        }
    } else
        fd = skfd;

    safe_strncpy(ifr.ifr_name, ifname, IFNAMSIZ);
    if (ioctl(fd, SIOCGIFFLAGS, &ifr) < 0) {
        fprintf(stderr, _("%s: ERROR while testing interface flags: %s\n"),
            ifname, strerror(errno));
        return -1;
    }
}

```

```

    return (ifr.ifr_flags & flags);
}

static void usage(void)
{
    fprintf(stderr, _("Usage:\n ifconfig [-a] [-v] [-s] <interface> [[<AF>] <address>]\n"));
#ifdef HAVE_AF_INET
    fprintf(stderr, _(" [add <address>[/<prefixlen>]]\n"));
    fprintf(stderr, _(" [del <address>[/<prefixlen>]]\n"));
    fprintf(stderr, _(" [[-]broadcast [<address>]] [[-]pointopoint [<address>]]\n"));
    fprintf(stderr, _(" [netmask <address>] [dstaddr <address>] [tunnel <address>]\n"));
#endif
#ifdef SIOCSKEEPALIVE
    fprintf(stderr, _(" [outfill <NN>] [keepalive <NN>]\n"));
#endif
    fprintf(stderr, _(" [hw <HW> <address>] [metric <NN>] [mtu <NN>]\n"));
    fprintf(stderr, _(" [[-]trailers] [[-]arp] [[-]allmulti]\n"));
    fprintf(stderr, _(" [multicast] [[-]promise]\n"));
    fprintf(stderr, _(" [mem_start <NN>] [io_addr <NN>] [irq <NN>] [media <type>]\n"));
#ifdef HAVE_TXQUEUELEN
    fprintf(stderr, _(" [txqueuelen <NN>]\n"));
#endif
#ifdef HAVE_DYNAMIC
    fprintf(stderr, _(" [[-]dynamic]\n"));
#endif
    fprintf(stderr, _(" [up|down] ...\n\n"));

    fprintf(stderr, _(" <HW>=Hardware Type.\n"));
    fprintf(stderr, _(" List of possible hardware types:\n"));
    print_hwlist(0); /* 1 = ARPable */
    fprintf(stderr, _(" <AF>=Address family. Default: %s\n"), DFLT_AF);
    fprintf(stderr, _(" List of possible address families:\n"));
    print_aflist(0); /* 1 = routeable */
    exit(E_USAGE);
}

static void version(void)
{
    fprintf(stderr, "%s\n%s\n", Release, Version);
    exit(E_USAGE);
}

static int set_netmask(int skfd, struct ifreq *ifr, struct sockaddr *sa)
{
    int err = 0;

    memcpy((char *) &ifr->ifr_netmask, (char *) sa,
           sizeof(struct sockaddr));
    if (ioctl(skfd, SIOCSIFNETMASK, ifr) < 0) {
        fprintf(stderr, "SIOCSIFNETMASK: %s\n",
                strerror(errno));
        err = 1;
    }
    return err;
}

```

```

}

int main(int argc, char **argv)
{
    struct sockaddr sa;
    struct sockaddr samask;
    struct sockaddr_in sin;
    char host[128];
    struct aftype *ap;
    struct hwtype *hw;
    struct ifreq ifr;
    int goterr = 0, didnetmask = 0, neednetmask=0;
    char **spp;
    int fd;
#ifdef HAVE_AF_INET6
    extern struct aftype inet6_aftype;
    struct sockaddr_in6 sa6;
    struct in6_ifreq ifr6;
    unsigned long prefix_len;
    char *cp;
#endif
#ifdef HAVE_AF_INET
    extern struct aftype inet_aftype;
#endif

#ifdef I18N
    setlocale (LC_ALL, "");
    bindtextdomain("net-tools", "/usr/share/locale");
    textdomain("net-tools");
#endif

    /* Find any options. */
    argc--;
    argv++;
    while (argc && *argv[0] == '-') {
        if (!strcmp(*argv, "-a"))
            opt_a = 1;

        else if (!strcmp(*argv, "-s"))
            ife_short = 1;

        else if (!strcmp(*argv, "-v"))
            opt_v = 1;

        else if (!strcmp(*argv, "-V") || !strcmp(*argv, "-version") ||
            !strcmp(*argv, "--version"))
            version();

        else if (!strcmp(*argv, "-?") || !strcmp(*argv, "-h") ||
            !strcmp(*argv, "-help") || !strcmp(*argv, "--help"))
            usage();

        else {
            fprintf(stderr, _("ifconfig: option '%s' not recognised.\n"),
                argv[0]);

```



```

        fprintf(stderr, _("ifconfig: `--help' gives usage information.\n"));
        exit(1);
    }

    argv++;
    argc--;
    }

    /* Create a channel to the NET kernel. */
    if ((skfd = sockets_open(0)) < 0) {
perror("socket");
exit(1);
    }

    /* Do we have to show the current setup? */
    if (argc == 0) {
int err = if_print((char *) NULL);
(void) close(skfd);
exit(err < 0);
    }
    /* No. Fetch the interface name. */
    spp = argv;
    safe_strncpy(ifr.ifr_name, *spp++, IFNAMSIZ);
    if (*spp == (char *) NULL) {
int err = if_print(ifr.ifr_name);
(void) close(skfd);
exit(err < 0);
    }

    /* The next argument is either an address family name, or an option. */
    if ((ap = get_aftype(*spp)) != NULL)
spp++; /* it was a AF name */
    else
ap = get_aftype(DFLT_AF);

    if (ap) {
addr_family = ap->af;
skfd = ap->fd;
    }

    /* Process the remaining arguments. */
    while (*spp != (char *) NULL) {
if (!strcmp(*spp, "arp")) {
goterr |= clr_flag(ifr.ifr_name, IFF_NOARP);
spp++;
continue;
}
if (!strcmp(*spp, "-arp")) {
goterr |= set_flag(ifr.ifr_name, IFF_NOARP);
spp++;
continue;
}
#ifdef IFF_PORTSEL
if (!strcmp(*spp, "media") || !strcmp(*spp, "port")) {

```

```

        if (*++spp == NULL)
        usage();
        if (!strcasecmp(*spp, "auto")) {
        goterr |= set_flag(ifr.ifr_name, IFF_AUTOMEDIA);
        } else {
        int i, j, newport;
        char *endp;
        newport = strtol(*spp, &endp, 10);
        if (*endp != 0) {
            newport = -1;
            for (i = 0; ifr.ifr_name[i][0] && newport == -1; i++) {
            for (j = 0; ifr.ifr_name[i][j]; j++) {
                if (!strcasecmp(*spp, ifr.ifr_name[i][j])) {
                newport = i;
                break;
                }
            }
            }
        }
        spp++;
        if (newport == -1) {
            fprintf(stderr, _("Unknown media type.\n"));
            goterr = 1;
        } else {
            if (ioctl(skfd, SIOCGIFMAP, &ifr) < 0) {
            perror("port: SIOCGIFMAP");
            goterr = 1;
            continue;
            }
            ifr.ifr_map.port = newport;
            if (ioctl(skfd, SIOCSIFMAP, &ifr) < 0) {
            perror("port: SIOCSIFMAP");
            goterr = 1;
            }
        }
        continue;
        }
    }
#endif

    if (!strcmp(*spp, "trailers")) {
        goterr |= clr_flag(ifr.ifr_name, IFF_NOTRAILERS);
        spp++;
        continue;
    }
    if (!strcmp(*spp, "-trailers")) {
        goterr |= set_flag(ifr.ifr_name, IFF_NOTRAILERS);
        spp++;
        continue;
    }
    if (!strcmp(*spp, "promisc")) {
        goterr |= set_flag(ifr.ifr_name, IFF_PROMISC);
        spp++;
        continue;
    }
}

```

```

if (!strcmp(*spp, "-promisc")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_PROMISC);
    if (test_flag(ifr.ifr_name, IFF_PROMISC) > 0)
        fprintf(stderr, _("Warning: Interface %s still in promisc mode... maybe other application is running?\n"),
ifr.ifr_name);
    spp++;
    continue;
}
if (!strcmp(*spp, "multicast")) {
    goterr |= set_flag(ifr.ifr_name, IFF_MULTICAST);
    spp++;
    continue;
}
if (!strcmp(*spp, "-multicast")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_MULTICAST);
    if (test_flag(ifr.ifr_name, IFF_MULTICAST) > 0)
        fprintf(stderr, _("Warning: Interface %s still in MULTICAST mode.\n"), ifr.ifr_name);
    spp++;
    continue;
}
if (!strcmp(*spp, "allmulti")) {
    goterr |= set_flag(ifr.ifr_name, IFF_ALLMULTI);
    spp++;
    continue;
}
if (!strcmp(*spp, "-allmulti")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_ALLMULTI);
    if (test_flag(ifr.ifr_name, IFF_MULTICAST) > 0)
        fprintf(stderr, _("Warning: Interface %s still in ALLMULTI mode.\n"), ifr.ifr_name);
    spp++;
    continue;
}
if (!strcmp(*spp, "up")) {
    goterr |= set_flag(ifr.ifr_name, (IFF_UP | IFF_RUNNING));
    spp++;
    continue;
}
if (!strcmp(*spp, "down")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_UP);
    spp++;
    continue;
}
#ifdef HAVE_DYNAMIC
if (!strcmp(*spp, "dynamic")) {
    goterr |= set_flag(ifr.ifr_name, IFF_DYNAMIC);
    spp++;
    continue;
}
if (!strcmp(*spp, "-dynamic")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_DYNAMIC);
    spp++;
    if (test_flag(ifr.ifr_name, IFF_MULTICAST) > 0)
        fprintf(stderr, _("Warning: Interface %s still in DYNAMIC mode.\n"), ifr.ifr_name);
    continue;
}

```

```

    }
#endif

    if (!strcmp(*spp, "metric")) {
        if (*++spp == NULL)
            usage();
        ifr.ifr_metric = atoi(*spp);
        if (ioctl(skfd, SIOCSIFMETRIC, &ifr) < 0) {
            fprintf(stderr, "SIOCSIFMETRIC: %s\n", strerror(errno));
            goterr = 1;
        }
        spp++;
        continue;
    }
    if (!strcmp(*spp, "mtu")) {
        if (*++spp == NULL)
            usage();
        ifr.ifr_mtu = atoi(*spp);
        if (ioctl(skfd, SIOCSIFMTU, &ifr) < 0) {
            fprintf(stderr, "SIOCSIFMTU: %s\n", strerror(errno));
            goterr = 1;
        }
        spp++;
        continue;
    }
#ifdef SIOCSKEEPALIVE
    if (!strcmp(*spp, "keepalive")) {
        if (*++spp == NULL)
            usage();
        ifr.ifr_data = (caddr_t) atoi(*spp);
        if (ioctl(skfd, SIOCSKEEPALIVE, &ifr) < 0) {
            fprintf(stderr, "SIOCSKEEPALIVE: %s\n", strerror(errno));
            goterr = 1;
        }
        spp++;
        continue;
    }
#endif

#ifdef SIOCSOUTFILL
    if (!strcmp(*spp, "outfill")) {
        if (*++spp == NULL)
            usage();
        ifr.ifr_data = (caddr_t) atoi(*spp);
        if (ioctl(skfd, SIOCSOUTFILL, &ifr) < 0) {
            fprintf(stderr, "SIOCSOUTFILL: %s\n", strerror(errno));
            goterr = 1;
        }
        spp++;
        continue;
    }
#endif

    if (!strcmp(*spp, "-broadcast")) {
        goterr |= clr_flag(ifr.ifr_name, IFF_BROADCAST);
    }

```

```

        if (test_flag(ifr.ifr_name, IFF_MULTICAST) > 0)
            fprintf(stderr, _("Warning: Interface %s still in BROADCAST mode.\n"), ifr.ifr_name);
        spp++;
        continue;
    }
    if (!strcmp(*spp, "broadcast")) {
        if (*++spp != NULL) {
            safe_strncpy(host, *spp, (sizeof host));
            if (ap->input(0, host, &sa) < 0) {
                if (ap->herror)
                    ap->herror(host);
                else
                    fprintf(stderr, _("ifconfig: Error resolving '%s' for broadcast\n"), host);
                goterr = 1;
                spp++;
                continue;
            }
            memcpy((char *) &ifr.ifr_broadaddr, (char *) &sa,
                sizeof(struct sockaddr));
            if (ioctl(ap->fd, SIOCSIFBRDADDR, &ifr) < 0) {
                fprintf(stderr, "SIOCSIFBRDADDR: %s\n",
                    strerror(errno));
                goterr = 1;
            }
        }
        spp++;
    }
    goterr |= set_flag(ifr.ifr_name, IFF_BROADCAST);
    continue;
}
if (!strcmp(*spp, "dstaddr")) {
    if (*++spp == NULL)
        usage();
    safe_strncpy(host, *spp, (sizeof host));
    if (ap->input(0, host, &sa) < 0) {
        if (ap->herror)
            ap->herror(host);
        else
            fprintf(stderr, _("ifconfig: Error resolving '%s' for dstaddr\n"), host);
        goterr = 1;
        spp++;
        continue;
    }
    memcpy((char *) &ifr.ifr_dstaddr, (char *) &sa,
        sizeof(struct sockaddr));
    if (ioctl(ap->fd, SIOCSIFDSTADDR, &ifr) < 0) {
        fprintf(stderr, "SIOCSIFDSTADDR: %s\n",
            strerror(errno));
        goterr = 1;
    }
    spp++;
    continue;
}
if (!strcmp(*spp, "netmask")) {
    if (*++spp == NULL || !didnetmask)

```

```

usage();
safe_strncpy(host, *spp, (sizeof host));
if (ap->input(0, host, &sa) < 0) {
    if (ap->herror)
        ap->herror(host);
    else
        fprintf(stderr, _("ifconfig: Error resolving '%s' for netmask\n"), host);
goterr = 1;
spp++;
continue;
}
didnetmask++;
goterr |= set_netmask(ap->fd, &ifr, &sa);
spp++;
continue;
}
#ifdef HAVE_TXQUEUELEN
if (!strcmp(*spp, "txqueuelen")) {
    if (*++spp == NULL)
usage();
    ifr.ifr_qlen = strtoul(*spp, NULL, 0);
    if (ioctl(skfd, SIOCSIFTXQLEN, &ifr) < 0) {
fprintf(stderr, "SIOCSIFTXQLEN: %s\n", strerror(errno));
goterr = 1;
    }
    spp++;
    continue;
}
#endif

if (!strcmp(*spp, "mem_start")) {
    if (*++spp == NULL)
usage();
    if (ioctl(skfd, SIOCGIFMAP, &ifr) < 0) {
fprintf(stderr, "mem_start: SIOCGIFMAP: %s\n", strerror(errno));
spp++;
goterr = 1;
continue;
    }
    ifr.ifr_map.mem_start = strtoul(*spp, NULL, 0);
    if (ioctl(skfd, SIOCSIFMAP, &ifr) < 0) {
fprintf(stderr, "mem_start: SIOCSIFMAP: %s\n", strerror(errno));
goterr = 1;
    }
    spp++;
    continue;
}

if (!strcmp(*spp, "io_addr")) {
    if (*++spp == NULL)
usage();
    if (ioctl(skfd, SIOCGIFMAP, &ifr) < 0) {
fprintf(stderr, "io_addr: SIOCGIFMAP: %s\n", strerror(errno));
spp++;
goterr = 1;
continue;
    }
}

```

```

    }
    ifr.ifr_map.base_addr = strtol(*spp, NULL, 0);
    if (ioctl(skfd, SIOCSIFMAP, &ifr) < 0) {
fprintf(stderr, "io_addr: SIOCSIFMAP: %s\n", strerror(errno));
goterr = 1;
    }
    spp++;
    continue;
}
if (!strcmp(*spp, "irq")) {
    if (*++spp == NULL)
usage();
    if (ioctl(skfd, SIOCGIFMAP, &ifr) < 0) {
fprintf(stderr, "irq: SIOCGIFMAP: %s\n", strerror(errno));
goterr = 1;
    spp++;
    continue;
    }
    ifr.ifr_map.irq = atoi(*spp);
    if (ioctl(skfd, SIOCSIFMAP, &ifr) < 0) {
fprintf(stderr, "irq: SIOCSIFMAP: %s\n", strerror(errno));
goterr = 1;
    }
    spp++;
    continue;
}
if (!strcmp(*spp, "-pointopoint")) {
    goterr |= clr_flag(ifr.ifr_name, IFF_POINTOPOINT);
    spp++;
    if (test_flag(ifr.ifr_name, IFF_MULTICAST) > 0)
fprintf(stderr, _("Warning: Interface %s still in POINTOPOINT mode.\n"), ifr.ifr_name);
    continue;
}
if (!strcmp(*spp, "pointopoint")) {
    if (*(spp + 1) != NULL) {
spp++;
safe_strncpy(host, *spp, (sizeof host));
if (ap->input(0, host, &sa)) {
    if (ap->herror)
        ap->herror(host);
    else
        fprintf(stderr, _("ifconfig: Error resolving '%s' for pointopoint\n"), host);
    goterr = 1;
    spp++;
    continue;
}
memcpy((char *) &ifr.ifr_dstaddr, (char *) &sa,
        sizeof(struct sockaddr));
if (ioctl(ap->fd, SIOCSIFDSTADDR, &ifr) < 0) {
    fprintf(stderr, "SIOCSIFDSTADDR: %s\n",
            strerror(errno));
    goterr = 1;
}
}
}

```

```

        goterr |= set_flag(ifr.ifr_name, IFF_POINTOPOINT);
        spp++;
        continue;
    };

    if (!strcmp(*spp, "hw")) {
        if (*++spp == NULL)
            usage();
        if ((hw = get_hwtype(*spp)) == NULL)
            usage();
        if (hw->input == NULL) {
            fprintf(stderr, _("hw address type '%s' has no handler to set address. failed.\n"), *spp);
            spp+=2;
            goterr = 1;
            continue;
        }
        if (*++spp == NULL)
            usage();
        safe_strncpy(host, *spp, (sizeof host));
        if (hw->input(host, &sa) < 0) {
            fprintf(stderr, _("%s: invalid %s address.\n"), host, hw->name);
            goterr = 1;
            spp++;
            continue;
        }
        memcpy((char *) &ifr.ifr_hwaddr, (char *) &sa,
            sizeof(struct sockaddr));
        if (ioctl(skfd, SIOCSIFHWADDR, &ifr) < 0) {
            if (errno == EBUSY)
                fprintf(stderr, "SIOCSIFHWADDR: %s - you may need to down the interface\n",
                    strerror(errno));
            else
                fprintf(stderr, "SIOCSIFHWADDR: %s\n",
                    strerror(errno));
            goterr = 1;
        }
        spp++;
        continue;
    }

#ifdef HAVE_AF_INET || HAVE_AF_INET6
    if (!strcmp(*spp, "add")) {
        if (*++spp == NULL)
            usage();
#ifdef HAVE_AF_INET6
        if (strchr(*spp, ':')) {
            /* INET6 */
            if ((cp = strchr(*spp, '/')) {
                prefix_len = atol(cp + 1);
                if ((prefix_len < 0) || (prefix_len > 128))
                    usage();
                *cp = 0;
            } else {
                prefix_len = 128;
            }
        }
        safe_strncpy(host, *spp, (sizeof host));

```



```

if (inet6_aftype.input(1, host,
    (struct sockaddr *) &sa6) < 0) {
    if (inet6_aftype.herror)
        inet6_aftype.herror(host);
    else
        fprintf(stderr, _("ifconfig: Error resolving '%s' for add\n"), host);
    goterr = 1;
    spp++;
    continue;
}
memcpy((char *) &ifr6.ifr6_addr, (char *) &sa6.sin6_addr,
    sizeof(struct in6_addr));

fd = get_socket_for_af(AF_INET6);
if (fd < 0) {
    fprintf(stderr,
        _("No support for INET6 on this system.\n"));
    goterr = 1;
    spp++;
    continue;
}
if (ioctl(fd, SIOGIFINDEX, &ifr) < 0) {
    perror("SIOGIFINDEX");
    goterr = 1;
    spp++;
    continue;
}
ifr6.ifr6_ifindex = ifr.ifr_ifindex;
ifr6.ifr6_prefixlen = prefix_len;
if (ioctl(fd, SIOCSIFADDR, &ifr6) < 0) {
    perror("SIOCSIFADDR");
    goterr = 1;
}
spp++;
continue;
}
#endif
#ifdef HAVE_AF_INET
    { /* ipv4 address a.b.c.d */
        unsigned long ip, nm, bc;
        safe_strncpy(host, *spp, (sizeof host));
        if (inet_aftype.input(0, host, (struct sockaddr *)&sin) < 0) {
            ap->herror(host);
            goterr = 1;
            spp++;
            continue;
        }
        fd = get_socket_for_af(AF_INET);
        if (fd < 0) {
            fprintf(stderr,
                _("No support for INET on this system.\n"));
            goterr = 1;
            spp++;
            continue;
        }
    }
}

```

```

    }

    memcpy(&ip, &sin.sin_addr.s_addr, sizeof(unsigned long));

    if (get_nmbc_parent(ifr.ifr_name, &nm, &bc) < 0) {
        fprintf(stderr, _("Interface %s not initialized\n"),
            ifr.ifr_name);
        goterr = 1;
        spp++;
        continue;
    }
    set_ifstate(ifr.ifr_name, ip, nm, bc, 1);

    }
    spp++;
    continue;
#else
    fprintf(stderr, _("Bad address.\n"));
#endif
}
#endif

#ifdef HAVE_AF_INET6 || HAVE_AF_INET6
if (!strcmp(*spp, "del")) {
    if (*++spp == NULL)
        usage();

#ifdef SIOC_DIFADDR
#ifdef HAVE_AF_INET6
        if (strchr(*spp, ':')) { /* INET6 */
            if ((cp = strchr(*spp, '/')) {
                prefix_len = atol(cp + 1);
                if ((prefix_len < 0) || (prefix_len > 128))
                    usage();
                *cp = 0;
            } else {
                prefix_len = 128;
            }
        }
        safe_strncpy(host, *spp, (sizeof host));
        if (inet6_atype.input(1, host,
            (struct sockaddr *) &sa6) < 0) {
            inet6_atype.herror(host);
            goterr = 1;
            spp++;
            continue;
        }
        memcpy((char *) &ifr6.ifr6_addr, (char *) &sa6.sin6_addr,
            sizeof(struct in6_addr));

        fd = get_socket_for_af(AF_INET6);
        if (fd < 0) {
            fprintf(stderr,
                _("No support for INET6 on this system.\n"));
            goterr = 1;
            spp++;
        }
    }
}

```

```

        continue;
    }
    if (ioctl(fd, SIOGIFINDEX, &ifr) < 0) {
        perror("SIOGIFINDEX");
        goterr = 1;
        spp++;
        continue;
    }
    ifr6.ifr6_ifindex = ifr.ifr_ifindex;
    ifr6.ifr6_prefixlen = prefix_len;
    if (opt_v)
        fprintf(stderr, "now deleting: ioctl(SIOCDIFADDR, {ifindex=%d, prefixlen=%ld})\n", ifr.ifr_ifindex, prefix_len);
    if (ioctl(fd, SIOCDIFADDR, &ifr6) < 0) {
        fprintf(stderr, "SIOCDIFADDR: %s\n",
            strerror(errno));
        goterr = 1;
    }
    spp++;
    continue;
}
#endif
#ifdef HAVE_AF_INET
{
    /* ipv4 address a.b.c.d */
    unsigned long ip, nm, bc;
    safe_strncpy(host, *spp, (sizeof host));
    if (inet_atype.input(0, host, (struct sockaddr *)&sin) < 0) {
        ap->herror(host);
        goterr = 1;
        spp++;
        continue;
    }
    fd = get_socket_for_af(AF_INET);
    if (fd < 0) {
        fprintf(stderr, _("No support for INET on this system.\n"));
        goterr = 1;
        spp++;
        continue;
    }

    /* Clear "ip" in case sizeof(unsigned long) > sizeof(sin.sin_addr.s_addr) */
    ip = 0;
    memcpy(&ip, &sin.sin_addr.s_addr, sizeof(sin.sin_addr.s_addr));

    if (get_nmbc_parent(ifr.ifr_name, &nm, &bc) < 0) {
        fprintf(stderr, _("Interface %s not initialized\n"),
            ifr.ifr_name);
        goterr = 1;
        spp++;
        continue;
    }
    set_ifstate(ifr.ifr_name, ip, nm, bc, 0);
}
    spp++;

```

```

        continue;
    #else
        fprintf(stderr, _("Bad address.\n"));
    #endif
    #else
        fprintf(stderr, _("Address deletion not supported on this system.\n"));
    #endif
}
#endif
#if HAVE_AF_INET6
if (!strcmp(*spp, "tunnel")) {
    if (*++spp == NULL)
        usage();
    if ((cp = strchr(*spp, '/')) {
        prefix_len = atol(cp + 1);
        if ((prefix_len < 0) || (prefix_len > 128))
            usage();
        *cp = 0;
    } else {
        prefix_len = 128;
    }
    safe_strncpy(host, *spp, (sizeof host));
    if (inet6_atype.input(1, host, (struct sockaddr *) &sa6) < 0) {
        inet6_atype.herror(host);
        goterr = 1;
        spp++;
        continue;
    }
    memcpy((char *) &ifr6.ifr6_addr, (char *) &sa6.sin6_addr,
        sizeof(struct in6_addr));

    fd = get_socket_for_af(AF_INET6);
    if (fd < 0) {
        fprintf(stderr, _("No support for INET6 on this system.\n"));
        goterr = 1;
        spp++;
        continue;
    }
    if (ioctl(fd, SIOGIFINDEX, &ifr) < 0) {
        perror("SIOGIFINDEX");
        goterr = 1;
        spp++;
        continue;
    }
    ifr6.ifr6_ifindex = ifr.ifr_ifindex;
    ifr6.ifr6_prefixlen = prefix_len;

    if (ioctl(fd, SIOCSIFDSTADDR, &ifr6) < 0) {
        fprintf(stderr, "SIOCSIFDSTADDR: %s\n",
            strerror(errno));
        goterr = 1;
    }
    spp++;
    continue;
}
}

```

```

#endif

/* If the next argument is a valid hostname, assume OK. */
safe_strncpy(host, *spp, (sizeof host));

/* FIXME: sa is too small for INET6 addresses, inet6 should use that too,
broadcast is unexpected */
if (ap->getmask) {
    switch (ap->getmask(host, &samask, NULL)) {
        case -1:
            usage();
            break;
        case 1:
            if (didnetmask)
                usage();

            // remeber to set the netmask from samask later
            neednetmask = 1;
            break;
    }
}
if (ap->input == NULL) {
    fprintf(stderr, _("ifconfig: Cannot set address for this protocol family.\n"));
    exit(1);
}
if (ap->input(0, host, &sa) < 0) {
    if (ap->herror)
        ap->herror(host);
    else
        fprintf(stderr, _("ifconfig: error resolving '%s' to set address for af=%s\n"), host, ap->name);
    fprintf(stderr, _("ifconfig: `--help' gives usage information.\n"));
    exit(1);
}
memcpy((char *) &ifr.ifr_addr, (char *) &sa, sizeof(struct sockaddr));
{
    int r = 0; /* to shut gcc up */
    switch (ap->af) {
#ifdef HAVE_AF_INET
        case AF_INET:
            fd = get_socket_for_af(AF_INET);
            if (fd < 0) {
                fprintf(stderr, _("No support for INET on this system.\n"));
                exit(1);
            }
            r = ioctl(fd, SIOCSIFADDR, &ifr);
            break;
#endif
#ifdef HAVE_AF_ECONET
        case AF_ECONET:
            fd = get_socket_for_af(AF_ECONET);
            if (fd < 0) {
                fprintf(stderr, _("No support for ECONET on this system.\n"));
                exit(1);
            }
            r = ioctl(fd, SIOCSIFADDR, &ifr);

```

```

break;
#endif
    default:
fprintf(stderr,
_("Don't know how to set addresses for family %d.\n"), ap->af);
exit(1);
    }
    if (r < 0) {
perror("SIOCSIFADDR");
goterr = 1;
    }
}

/*
 * Don't do the set_flag() if the address is an alias with a - at the
 * end, since it's deleted already! - Roman
 *
 * Should really use regex.h here, not sure though how well it'll go
 * with the cross-platform support etc.
 */
{
    char *ptr;
    short int found_colon = 0;
    for (ptr = ifr.ifr_name; *ptr; ptr++)
        if (*ptr == ':') found_colon++;

    if (!(found_colon && *(ptr - 1) == '-'))
        goterr |= set_flag(ifr.ifr_name, (IFF_UP | IFF_RUNNING));
}

spp++;
}

if (neednetmask) {
goterr |= set_netmask(skfd, &ifr, &samask);
didnetmask++;
}

if (opt_v && goterr)
fprintf(stderr, _("WARNING: at least one error occured. (%d)\n"), goterr);

return (goterr);
}

struct ifcmd {
    int flag;
    unsigned long addr;
    char *base;
    int baselen;
};

static unsigned char searcher[256];

static int set_ip_using(const char *name, int c, unsigned long ip)
{

```

```

    struct ifreq ifr;
    struct sockaddr_in sin;

    safe_strncpy(ifr.ifr_name, name, IFNAMSIZ);
    memset(&sin, 0, sizeof(struct sockaddr));
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = ip;
    memcpy(&ifr.ifr_addr, &sin, sizeof(struct sockaddr));
    if (ioctl(skfd, c, &ifr) < 0)
return -1;
    return 0;
}

static int do_ifcmd(struct interface *x, struct ifcmd *ptr)
{
    char *z, *e;
    struct sockaddr_in *sin;
    int i;

    if (do_if_fetch(x) < 0)
return 0;
    if (strcmp(x->name, ptr->base, ptr->baselen) != 0)
return 0; /* skip */
    z = strchr(x->name, '.');
    if (!z || !*z)
return 0;
    z++;
    for (e = z; *e; e++)
if (*e == '.') /* deleted */
    return 0;
    i = atoi(z);
    if (i < 0 || i > 255)
abort();
    searcher[i] = 1;

    /* copy */
    sin = (struct sockaddr_in *)&x->dstaddr;
    if (sin->sin_addr.s_addr != ptr->addr) {
return 0;
    }

    if (ptr->flag) {
/* turn UP */
if (set_flag(x->name, IFF_UP | IFF_RUNNING) == -1)
    return -1;
    } else {
/* turn DOWN */
if (clr_flag(x->name, IFF_UP) == -1)
    return -1;
    }

    return 1; /* all done! */
}

```

```

static int get_nmbc_parent(char *parent,
    unsigned long *nm, unsigned long *bc)
{
    struct interface *i;
    struct sockaddr_in *sin;

    i = lookup_interface(parent);
    if (!i)
return -1;
    if (do_if_fetch(i) < 0)
return 0;
    sin = (struct sockaddr_in *)&i->netmask;
    memcpy(nm, &sin->sin_addr.s_addr, sizeof(unsigned long));
    sin = (struct sockaddr_in *)&i->broadaddr;
    memcpy(bc, &sin->sin_addr.s_addr, sizeof(unsigned long));
    return 0;
}

static int set_ifstate(char *parent, unsigned long ip,
    unsigned long nm, unsigned long bc,
    int flag)
{
    char buf[IFNAMSIZ];
    struct ifcmd pt;
    int i;

    pt.base = parent;
    pt.baselen = strlen(parent);
    pt.addr = ip;
    pt.flag = flag;
    memset(searcher, 0, sizeof(searcher));
    i = for_all_interfaces((int (*)(struct interface *, void *))do_ifcmd,
    &pt);
    if (i == -1)
return -1;
    if (i == 1)
return 0;

    /* add a new interface */
    for (i = 0; i < 256; i++)
if (searcher[i] == 0)
    break;

    if (i == 256)
return -1; /* FAILURE!!! out of ip addresses */

    if (snprintf(buf, IFNAMSIZ, "%s:%d", parent, i) > IFNAMSIZ)
return -1;
    if (set_ip_using(buf, SIOCSIFADDR, ip) == -1)
return -1;
    if (set_ip_using(buf, SIOCSIFNETMASK, nm) == -1)
return -1;
    if (set_ip_using(buf, SIOCSIFBRDADDR, bc) == -1)
return -1;

```



```
        if (set_flag(buf, IFF_BROADCAST) == -1)
return -1;
        return 0;
}
```

1.97 netbase

1.97.1 Available under license

This package was created by Peter Tobias tobias@et-inf.fho-emden.de on Wed, 24 Aug 1994 21:33:28 +0200 and maintained by Anthony Towns <ajt@debian.org> until 2001.
It is currently maintained by Marco d'Itri <md@linux.it>.

Copyright 1994-2010 Peter Tobias, Anthony Towns and Marco d'Itri

The programs in this package are distributed under the terms of the GNU General Public License, version 2 as distributed by the Free Software Foundation. On Debian systems, a copy of this license may be found in /usr/share/common-licenses/GPL-2.

1.98 netkit-rsh

1.98.1 Available under license

```

/*_
 * Copyright (c) 1983, 1990 The Regents of the University of California.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 * must display the following acknowledgement:
 * This product includes software developed by the University of
 * California, Berkeley and its contributors.
 * 4. Neither the name of the University nor the names of its contributors
 * may be used to endorse or promote products derived from this software
 * without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

char copyright[] =
"@(#) Copyright (c) 1983, 1990 The Regents of the University of California.\n"
"All rights reserved.\n";

/*
 * From: @(#)rsh.c5.24 (Berkeley) 7/1/91
 */
char rsid[] = "$Id: rsh.c,v 1.13 2000/07/23 04:16:24 dholland Exp $";
#include "../version.h"

#include <sys/types.h>
#include <signal.h>
#include <sys/socket.h>
#include <sys/ioctl.h>

```

```

#include <sys/file.h>
#include <sys/time.h>
#if defined(linux) && defined(FSUID_HACK)
#include <sys/fsuid.h>
#endif

#include <netinet/in.h>
#include <netdb.h>

#include <pwd.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include "pathnames.h"

/*
 * rsh - remote shell
 */
static int rfd2;
static char *copyargs(char **);
static void sendsig(int);
static void talk(int nflag, long omask, int pid, int rem);
static void usage(void);

int
main(int argc, char *argv[])
{
    struct passwd *pw;
    struct servent *sp;
    long omask;
    int argoff, asrsh, ch, dflag, nflag, one, pid=0, rem, uid;
    char *p;
    char *args, *host, *user;
    char *null = NULL;
    char **saved_environ;

    saved_environ = __environ;
    __environ = &null;

    argoff = asrsh = dflag = nflag = 0;
    one = 1;
    host = user = NULL;

    /* if called as something other than "rsh", use it as the host name */
    p = strchr(argv[0], '/');
    if (p) p++;
    else p = argv[0];

    #ifndef debian
    if (!strcmp(p, "rsh") || !strcmp(p, "netkit-rsh")) asrsh = 1;
    #else
    if (!strcmp(p, "rsh")) asrsh = 1;
    #endif

```

```
/* handle "rsh host flags" */
if (!host && argc > 2 && argv[1][0] != '-') {
    host = argv[1];
    argoff = 1;
}

#define OPTIONS "+8KLdel:nw"
while ((ch = getopt(argc - argoff, argv + argoff, OPTIONS)) != EOF)
    switch(ch) {
        case 'K':
            break;
        case 'L': /* -8Lew are ignored to allow rlogin aliases */
        case 'e':
        case 'w':
        case '8':
            break;
        case 'd':
            dflag = 1;
            break;
        case 'l':
            user = optarg;
            break;
        case 'n':
            nflag = 1;
            break;
        case '?':
        default:
            usage();
    }
    optind += argoff;

/* if haven't gotten a host yet, do so */
if (!host && !(host = argv[optind++]))
    usage();

/* if no further arguments, must have been called as rlogin. */
if (!argv[optind]) {
    if (setuid(getuid())) {
        fprintf(stderr, "rsh: setuid: %s\n", strerror(errno));
        exit(1);
    }
    if (asrsh) argv[0] = (char *) "rlogin";
    execve(_PATH_RLOGIN, argv, saved_environ);
    fprintf(stderr, "rsh: can't exec %s.\n", _PATH_RLOGIN);
    exit(1);
}

argc -= optind;
argv += optind;

if (!(pw = getpwuid(uid = getuid()))) {
    fprintf(stderr, "rsh: unknown user id.\n");
    exit(1);
}
```

```

    }
    if (!user)
        user = pw->pw_name;

    args = copyargs(argv);

    sp = NULL;
    if (sp == NULL)
        sp = getservbyname("shell", "tcp");
    if (sp == NULL) {
        fprintf(stderr, "rsh: shell/tcp: unknown service.\n");
        exit(1);
    }

    #if defined(linux) && defined(FSUID_HACK)
        setsuid(getuid());
    #endif
    rem = rcmd_af(&host, sp->s_port, pw->pw_name, user, args, &rfd2,
        AF_UNSPEC);
    #if defined(linux) && defined(FSUID_HACK)
        setsuid(geteuid());
    #endif

    if (rem < 0)
        exit(1);

    if (rfd2 < 0) {
        fprintf(stderr, "rsh: can't establish stderr.\n");
        exit(1);
    }

    if (setuid(uid)) {
        fprintf(stderr, "rsh: setuid: %s\n", strerror(errno));
        exit(1);
    }

    if (dflag) {
        if (setsockopt(rem, SOL_SOCKET, SO_DEBUG, &one,
            sizeof(one)) < 0)
            fprintf(stderr, "rsh: setsockopt: %s.\n",
                strerror(errno));
        if (setsockopt(rfd2, SOL_SOCKET, SO_DEBUG, &one,
            sizeof(one)) < 0)
            fprintf(stderr, "rsh: setsockopt: %s.\n",
                strerror(errno));
    }

    omask = sigblock(sigmask(SIGINT)|sigmask(SIGQUIT)|sigmask(SIGTERM));
    if (signal(SIGINT, SIG_IGN) != SIG_IGN)
        signal(SIGINT, sendsig);
    if (signal(SIGQUIT, SIG_IGN) != SIG_IGN)
        signal(SIGQUIT, sendsig);
    if (signal(SIGTERM, SIG_IGN) != SIG_IGN)
        signal(SIGTERM, sendsig);

```

```
if (!nflag) {
pid = fork();
if (pid < 0) {
fprintf(stderr,
    "rsh: fork: %s.\n", strerror(errno));
exit(1);
}
}

{
ioctl(rfd2, FIONBIO, &one);
ioctl(rem, FIONBIO, &one);
}

talk(nflag, omask, pid, rem);

if (!nflag)
kill(pid, SIGKILL);
exit(0);
}

static void
talk(int nflag, long omask, int pid, int rem)
{
register int cc, wc;
register char *bp;
fd_set readfrom, rembits;
int rfd2_ok, rem_ok;
char buf[BUFSIZ];

FD_ZERO(&rembits);

if (!nflag && pid == 0) {
close(rfd2);

reread:errno = 0;
if ((cc = read(0, buf, sizeof buf)) <= 0)
goto done;
bp = buf;

rewrite:FD_ZERO(&rembits);
FD_SET(rem, &rembits);
if (select(rem+1, 0, &rembits, 0, 0) < 0) {
if (errno != EINTR) {
fprintf(stderr,
    "rsh: select: %s.\n", strerror(errno));
exit(1);
}
goto rewrite;
}
if (!FD_ISSET(rem, &rembits))
goto rewrite;
wc = write(rem, bp, cc);
```

```

    if (wc < 0) {
    if (errno == EWOULDBLOCK)
    goto rewrite;
    goto done;
    }
    bp += wc;
    cc -= wc;
    if (cc == 0)
    goto reread;
    goto rewrite;
done:
shutdown(rem, 1);
exit(0);
}

rfd2_ok = rem_ok = 1;
sigsetmask(omask);
while (rfd2_ok || rem_ok) {
    FD_ZERO(&readfrom);
    if (rfd2_ok)
        FD_SET(rfd2, &readfrom);
    if (rem_ok)
        FD_SET(rem, &readfrom);
    if (select(rfd2 > rem ? rfd2+1 : rem+1,
        &readfrom, 0, 0, 0) < 0) {
        if (errno != EINTR) {
            fprintf(stderr,
                "rsh: select: %s.\n", strerror(errno));
            exit(1);
        }
        continue;
    }
    if (FD_ISSET(rfd2, &readfrom)) {
        errno = 0;
        cc = read(rfd2, buf, sizeof buf);
        if (cc > 0)
            write(2, buf, cc);
        else if (cc == 0 || errno != EWOULDBLOCK)
            rfd2_ok = 0;
    }
    if (FD_ISSET(rem, &readfrom)) {
        errno = 0;
        cc = read(rem, buf, sizeof buf);
        if (cc > 0)
            write(1, buf, cc);
        else if (cc == 0 || errno != EWOULDBLOCK)
            rem_ok = 0;
    }
}

void
sendsig(int signo)
{
    char x = (char) signo;

```



```
write(rfd2, &x, 1);
}

char *
copyargs(char **argv)
{
    int cc;
    char **ap, *p;
    char *args;

    cc = 0;
    for (ap = argv; *ap; ++ap)
        cc += strlen(*ap) + 1;
    args = malloc(cc);
    if (!args) {
        fprintf(stderr, "rsh: %s.\n", strerror(ENOMEM));
        exit(1);
    }
    for (p = args, ap = argv; *ap; ++ap) {
        /*strcpy(p, *ap);*/
        for (p = strcpy(p, *ap); *p; ++p);
        if (ap[1])
            *p++ = ' ';
    }
    return(args);
}

void
usage(void)
{
    fprintf(stderr,
        "usage: rsh [-nd%s] %s[-l login] host [command]\n",
        "", "");
    exit(1);
}
```

1.99 nettle

1.99.1 Available under license

GNU LESSER GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates
the terms and conditions of version 3 of the GNU General Public
License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser
General Public License, and the "GNU GPL" refers to version 3 of the GNU
General Public License.

"The Library" refers to a covered work governed by this License,
other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided
by the Library, but which is not otherwise based on the Library.
Defining a subclass of a class defined by the Library is deemed a mode
of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an
Application with the Library. The particular version of the Library
with which the Combined Work was made is also called the "Linked
Version".

The "Minimal Corresponding Source" for a Combined Work means the
Corresponding Source for the Combined Work, excluding any source code
for portions of the Combined Work that, considered in isolation, are
based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the
object code and/or source code for the Application, including any data
and utility programs needed for reproducing the Combined Work from the
Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License
without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form

suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser

General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so

that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for

making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under

any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

/* serpent-decrypt.c

The serpent block cipher.

For more details on this algorithm, see the Serpent website at
<http://www.cl.cam.ac.uk/~rja14/serpent.html>

Copyright (C) 2011 Niels M  ller
Copyright (C) 2010, 2011 Simon Josefsson
Copyright (C) 2003, 2004, 2005 Free Software Foundation, Inc.

This file is part of GNU Nettle.

GNU Nettle is free software: you can redistribute it and/or
modify it under the terms of either:

- * the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

or

- * the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

or both in parallel, as here.

GNU Nettle is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
General Public License for more details.

You should have received copies of the GNU General Public License and

```

the GNU Lesser General Public License along with this program. If
not, see http://www.gnu.org/licenses/.
*/

/* This file is derived from cipher/serpent.c in Libgcrypt v1.4.6.
The adaption to Nettle was made by Simon Josefsson on 2010-12-07
with final touches on 2011-05-30. Changes include replacing
libgcrypt with nettle in the license template, renaming
serpent_context to serpent_ctx, renaming u32 to uint32_t, removing
libgcrypt stubs and selftests, modifying entry function prototypes,
using FOR_BLOCKS to iterate through data in encrypt/decrypt, using
LE_READ_UINT32 and LE_WRITE_UINT32 to access data in
encrypt/decrypt, and running indent on the code. */

#if HAVE_CONFIG_H
#include "config.h"
#endif

#include <assert.h>
#include <limits.h>

#include "serpent.h"

#include "macros.h"
#include "serpent-internal.h"

/* These are the S-Boxes of Serpent. They are copied from Serpents
reference implementation (the optimized one, contained in
`floppy2') and are therefore:

Copyright (C) 1998 Ross Anderson, Eli Biham, Lars Knudsen.

To quote the Serpent homepage
(http://www.cl.cam.ac.uk/~rja14/serpent.html):

"Serpent is now completely in the public domain, and we impose no
restrictions on its use. This was announced on the 21st August at
the First AES Candidate Conference. The optimised implementations
in the submission package are now under the GNU PUBLIC LICENSE
(GPL), although some comments in the code still say otherwise. You
are welcome to use Serpent for any application." */

/* S0 inverse: 13 3 11 0 10 6 5 12 1 14 4 7 15 9 8 2 */
/* Original single-assignment form:

t01 = x2 ^ x3;
t02 = x0 | x1;
t03 = x1 | x2;
t04 = x2 & t01;
t05 = t02 ^ t01;
t06 = x0 | t04;
y2 = ~ t05;
t08 = x1 ^ x3;
t09 = t03 & t08;
t10 = x3 | y2;

```

```

    y1 = t09 ^ t06;
    t12 = x0 | t05;
    t13 = y1 ^ t12;
    t14 = t03 ^ t10;
    t15 = x0 ^ x2;
    y3 = t14 ^ t13;
    t17 = t05 & t13;
    t18 = t14 | t17;
    y0 = t15 ^ t18;
*/
#define SBOX0_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3)\
do {\
    y0 = x0 ^ x2;\
    y2 = x0 | x1;\
    y1 = x2 ^ x3;\
    y2 ^= y1;\
    y1 &= x2;\
    x2 |= x1;\
    x1 ^= x3;\
    y1 |= x0;\
    x1 &= x2;\
    y1 ^= x1;\
    x0 |= y2;\
    x0 ^= y1;\
    x1 = y2 & x0;\
    y2 = ~ y2;\
    x3 |= y2;\
    x3 ^= x2;\
    y3 = x3 ^ x0;\
    x1 |= x3;\
    y0 ^= x1;\
} while (0)

/* S1 inverse:  5 8 2 14 15 6 12 3 11 4 7 9 1 13 10 0 */
/* Original single-assignment form:
    t01 = x0 ^ x1;
    t02 = x1 | x3;
    t03 = x0 & x2;
    t04 = x2 ^ t02;
    t05 = x0 | t04;
    t06 = t01 & t05;
    t07 = x3 | t03;
    t08 = x1 ^ t06;
    t09 = t07 ^ t06;
    t10 = t04 | t03;
    t11 = x3 & t08;
    y2 = ~ t09;
    y1 = t10 ^ t11;
    t14 = x0 | y2;
    t15 = t06 ^ y1;
    y3 = t01 ^ t04;
    t17 = x2 ^ t15;
    y0 = t14 ^ t17;
*/

```

```

#define SBOX1_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
    y1 = x1 | x3; \
    y1 ^= x2; \
    y3 = x0 ^ x1; \
    y0 = x0 | y1; \
    y0 &= y3; \
    x1 ^= y0; \
    y3 ^= y1; \
    x1 &= x3; \
    y2 = x0 & x2; \
    y1 |= y2; \
    y2 |= x3; \
    y2 ^= y0; \
    y2 = ~ y2; \
    y1 ^= x1; \
    y0 ^= y1; \
    y0 ^= x2; \
    x0 |= y2; \
    y0 ^= x0; \
} while (0)

```

```

/* S2 inverse: 12 9 15 4 11 14 1 2 0 3 6 13 5 8 10 7 */

```

```

/* Original single-assignment form:

```

```

    t01 = x0 ^ x3;
    t02 = x2 ^ x3;
    t03 = x0 & x2;
    t04 = x1 | t02;
    y0 = t01 ^ t04;
    t06 = x0 | x2;
    t07 = x3 | y0;
    t08 = ~ x3;
    t09 = x1 & t06;
    t10 = t08 | t03;
    t11 = x1 & t07;
    t12 = t06 & t02;
    y3 = t09 ^ t10;
    y1 = t12 ^ t11;
    t15 = x2 & y3;
    t16 = y0 ^ y1;
    t17 = t10 ^ t15;
    y2 = t16 ^ t17;

```

```

*/

```

```

#define SBOX2_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3)\
do {\
    y0 = x0 ^ x3;\
    y2 = x2 ^ x3;\
    y1 = x1 | y2;\
    y0 ^= y1;\
    y1 = x3 | y0;\
    y1 &= x1;\
    x3 = ~ x3;\
    y3 = x0 | x2;\
    y2 &= y3;\
    y1 ^= y2;\
}

```

```

y3 &= x1;\
x0 &= x2;\
x0 |= x3;\
y3 ^= x0;\
x2 &= y3;\
x2 ^= x0;\
y2 = y0 ^ y1;\
y2 ^= x2;\
} while (0)

/* S3 inverse: 0 9 10 7 11 14 6 13 3 5 12 2 4 8 15 1 */
/* Original single-assignment form:
t01 = x2 | x3;
t02 = x0 | x3;
t03 = x2 ^ t02;
t04 = x1 ^ t02;
t05 = x0 ^ x3;
t06 = t04 & t03;
t07 = x1 & t01;
y2 = t05 ^ t06;
t09 = x0 ^ t03;
y0 = t07 ^ t03;
t11 = y0 | t05;
t12 = t09 & t11;
t13 = x0 & y2;
t14 = t01 ^ t05;
y1 = x1 ^ t12;
t16 = x1 | t13;
y3 = t14 ^ t16;
*/
#define SBOX3_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
y3 = x2 | x3; \
y0 = x1 & y3; \
y2 = x0 | x3; \
y1 = x2 ^ y2; \
y0 ^= y1; \
x3 ^= x0; \
y3 ^= x3; \
y2 ^= x1; \
y2 &= y1; \
y2 ^= x3; \
y1 ^= x0; \
x3 |= y0; \
y1 &= x3; \
y1 ^= x1; \
x0 &= y2; \
x0 |= x1; \
y3 ^= x0; \
} while (0)

/* S4 inverse: 5 0 8 3 10 9 7 14 2 12 11 6 4 15 13 1 */
/* Original single-assignment form:
t01 = x1 | x3;

```

```

t02 = x2 | x3;
t03 = x0 & t01;
t04 = x1 ^ t02;
t05 = x2 ^ x3;
t06 = ~ t03;
t07 = x0 & t04;
y1 = t05 ^ t07;
t09 = y1 | t06;
t10 = x0 ^ t07;
t11 = t01 ^ t09;
t12 = x3 ^ t04;
t13 = x2 | t10;
y3 = t03 ^ t12;
t15 = x0 ^ t04;
y2 = t11 ^ t13;
y0 = t15 ^ t09;
*/
#define SBOX4_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
    y1 = x2 ^ x3; \
    y2 = x2 | x3; \
    y2 ^= x1; \
    x1 |= x3; \
    y0 = x0 ^ y2; \
    x3 ^= y2; \
    y2 &= x0; \
    y1 ^= y2; \
    y2 ^= x0; \
    y2 |= x2; \
    x0 &= x1; \
    y3 = x0 ^ x3; \
    x0 = ~ x0; \
    x0 |= y1; \
    y0 ^= x0; \
    x0 ^= x1; \
    y2 ^= x0; \
} while (0)

/* S5 inverse: 8 15 2 9 4 1 13 14 11 6 5 3 7 12 10 0 */
/* Original single-assignment form:
t01 = x0 & x3;
t02 = x2 ^ t01;
t03 = x0 ^ x3;
t04 = x1 & t02;
t05 = x0 & x2;
y0 = t03 ^ t04;
t07 = x0 & y0;
t08 = t01 ^ y0;
t09 = x1 | t05;
t10 = ~ x1;
y1 = t08 ^ t09;
t12 = t10 | t07;
t13 = y0 | y1;
y3 = t02 ^ t12;
t15 = t02 ^ t13;

```



```

    t16 = x1 ^ x3;
    y2 = t16 ^ t15;
*/
#define SBOX5_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
    y1 = x0 & x3; \
    y3 = x2 ^ y1; \
    y0 = x1 & y3; \
    y2 = x0 ^ x3; \
    x3 ^= x1; \
    y0 ^= y2; \
    x2 &= x0; \
    x0 &= y0; \
    x2 |= x1; \
    y1 ^= y0; \
    y1 ^= x2; \
    y2 = y0 | y1; \
    y2 ^= y3; \
    y2 ^= x3; \
    x1 = ~ x1; \
    x1 |= x0; \
    y3 ^= x1; \
} while (0)

/* S6 inverse: 15 10 1 13 5 3 6 0 4 9 14 7 2 12 8 11 */
/* Original single-assignment form:
    t01 = x0 ^ x2;
    t02 = ~ x2;
    t03 = x1 & t01;
    t04 = x1 | t02;
    t05 = x3 | t03;
    t06 = x1 ^ x3;
    t07 = x0 & t04;
    t08 = x0 | t02;
    t09 = t07 ^ t05;
    y1 = t06 ^ t08;
    y0 = ~ t09;
    t12 = x1 & y0;
    t13 = t01 & t05;
    t14 = t01 ^ t12;
    t15 = t07 ^ t13;
    t16 = x3 | t02;
    t17 = x0 ^ y1;
    y3 = t17 ^ t15;
    y2 = t16 ^ t14;
*/
#define SBOX6_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
    y2 = x0 ^ x2; \
    x2 = ~ x2; \
    y0 = x1 ^ x3; \
    y1 = x0 | x2; \
    y1 ^= y0; \
    y3 = x1 & y2; \

```

```

y3 |= x3; \
x3 |= x2; \
x2 |= x1; \
x2 &= x0; \
y0 = x2 ^ y3; \
y0 = ~ y0; \
y3 &= y2; \
y3 ^= x2; \
x0 ^= y1; \
y3 ^= x0; \
x1 &= y0; \
y2 ^= x1; \
y2 ^= x3; \
} while (0)

/* S7 inverse: 3 0 6 13 9 14 15 8 5 12 11 7 10 1 4 2 */
/* Original single-assignment form:
t01 = x0 & x1;
t02 = x0 | x1;
t03 = x2 | t01;
t04 = x3 & t02;
y3 = t03 ^ t04;
t06 = x1 ^ t04;
t07 = x3 ^ y3;
t08 = ~ t07;
t09 = t06 | t08;
t10 = x1 ^ x3;
t11 = x0 | x3;
y1 = x0 ^ t09;
t13 = x2 ^ t06;
t14 = x2 & t11;
t15 = x3 | y1;
t16 = t01 | t10;
y0 = t13 ^ t15;
y2 = t14 ^ t16;
*/
#define SBOX7_INVERSE(x0, x1, x2, x3, y0, y1, y2, y3) \
do { \
y3 = x0 & x1; \
y2 = x1 ^ x3; \
y2 |= y3; \
y1 = x0 | x3; \
y1 &= x2; \
y2 ^= y1; \
y3 |= x2; \
y0 = x0 | x1; \
y0 &= x3; \
y3 ^= y0; \
y0 ^= x1; \
y1 = x3 ^ y3; \
y1 = ~ y1; \
y1 |= y0; \
y0 ^= x2; \
y1 ^= x0; \
x3 |= y1; \

```

```

    y0 ^= x3; \
} while (0)

/* In-place inverse linear transformation. */
#define LINEAR_TRANSFORMATION_INVERSE(x0,x1,x2,x3) \
do { \
    x2 = ROTL32 (10, x2); \
    x0 = ROTL32 (27, x0); \
    x2 = x2 ^ x3 ^ (x1 << 7); \
    x0 = x0 ^ x1 ^ x3; \
    x3 = ROTL32 (25, x3); \
    x1 = ROTL32 (31, x1); \
    x3 = x3 ^ x2 ^ (x0 << 3); \
    x1 = x1 ^ x0 ^ x2; \
    x2 = ROTL32 (29, x2); \
    x0 = ROTL32 (19, x0); \
} while (0)

/* Round inputs are x0,x1,x2,x3 (destroyed), and round outputs are
   y0,y1,y2,y3. */
#define ROUND_INVERSE(which, subkey, x0,x1,x2,x3, y0,y1,y2,y3) \
do { \
    LINEAR_TRANSFORMATION_INVERSE (x0,x1,x2,x3); \
    SBOX##which##_INVERSE(x0,x1,x2,x3, y0,y1,y2,y3); \
    KEYXOR(y0,y1,y2,y3, subkey); \
} while (0)

#if HAVE_NATIVE_64_BIT

/* In-place inverse linear transformation. */
#define LINEAR_TRANSFORMATION64_INVERSE(x0,x1,x2,x3) \
do { \
    x2 = DROTL32 (10, x2); \
    x0 = DROTL32 (27, x0); \
    x2 = x2 ^ x3 ^ DRSHIFT32(7, x1); \
    x0 = x0 ^ x1 ^ x3; \
    x3 = DROTL32 (25, x3); \
    x1 = DROTL32 (31, x1); \
    x3 = x3 ^ x2 ^ DRSHIFT32(3, x0); \
    x1 = x1 ^ x0 ^ x2; \
    x2 = DROTL32 (29, x2); \
    x0 = DROTL32 (19, x0); \
} while (0)

#define ROUND64_INVERSE(which, subkey, x0,x1,x2,x3, y0,y1,y2,y3) \
do { \
    LINEAR_TRANSFORMATION64_INVERSE (x0,x1,x2,x3); \
    SBOX##which##_INVERSE(x0,x1,x2,x3, y0,y1,y2,y3); \
    KEYXOR64(y0,y1,y2,y3, subkey); \
} while (0)

#endif /* HAVE_NATIVE_64_BIT */

void
```

```

serpent_decrypt(const struct serpent_ctx *ctx,
size_t length, uint8_t * dst, const uint8_t * src)
{
    assert( !(length % SERPENT_BLOCK_SIZE));

#ifdef HAVE_NATIVE_64_BIT
    if (length & SERPENT_BLOCK_SIZE)
#else
    while (length >= SERPENT_BLOCK_SIZE)
#endif
    {
        uint32_t x0,x1,x2,x3, y0,y1,y2,y3;
        unsigned k;

        x0 = LE_READ_UINT32(src);
        x1 = LE_READ_UINT32(src + 4);
        x2 = LE_READ_UINT32(src + 8);
        x3 = LE_READ_UINT32(src + 12);

        /* Inverse of special round */
        KEYXOR(x0,x1,x2,x3, ctx->keys[32]);
        SBOX7_INVERSE(x0,x1,x2,x3, y0,y1,y2,y3);
        KEYXOR(y0,y1,y2,y3, ctx->keys[31]);

        k = 24;
        goto start32;
        while (k > 0)
        {
            k -= 8;
            ROUND_INVERSE(7, ctx->keys[k+7], x0,x1,x2,x3, y0,y1,y2,y3);
start32:
            ROUND_INVERSE(6, ctx->keys[k+6], y0,y1,y2,y3, x0,x1,x2,x3);
            ROUND_INVERSE(5, ctx->keys[k+5], x0,x1,x2,x3, y0,y1,y2,y3);
            ROUND_INVERSE(4, ctx->keys[k+4], y0,y1,y2,y3, x0,x1,x2,x3);
            ROUND_INVERSE(3, ctx->keys[k+3], x0,x1,x2,x3, y0,y1,y2,y3);
            ROUND_INVERSE(2, ctx->keys[k+2], y0,y1,y2,y3, x0,x1,x2,x3);
            ROUND_INVERSE(1, ctx->keys[k+1], x0,x1,x2,x3, y0,y1,y2,y3);
            ROUND_INVERSE(0, ctx->keys[k], y0,y1,y2,y3, x0,x1,x2,x3);
        }

        LE_WRITE_UINT32(dst, x0);
        LE_WRITE_UINT32(dst + 4, x1);
        LE_WRITE_UINT32(dst + 8, x2);
        LE_WRITE_UINT32(dst + 12, x3);

        src += SERPENT_BLOCK_SIZE;
        dst += SERPENT_BLOCK_SIZE;
        length -= SERPENT_BLOCK_SIZE;
    }
#ifdef HAVE_NATIVE_64_BIT
    FOR_BLOCKS(length, dst, src, 2*SERPENT_BLOCK_SIZE)
    {
        uint64_t x0,x1,x2,x3, y0,y1,y2,y3;
        unsigned k;

```

```

x0 = LE_READ_UINT32 (src);
x1 = LE_READ_UINT32 (src + 4);
x2 = LE_READ_UINT32 (src + 8);
x3 = LE_READ_UINT32 (src + 12);

x0 <=<= 32; x0 |= LE_READ_UINT32 (src + 16);
x1 <=<= 32; x1 |= LE_READ_UINT32 (src + 20);
x2 <=<= 32; x2 |= LE_READ_UINT32 (src + 24);
x3 <=<= 32; x3 |= LE_READ_UINT32 (src + 28);

/* Inverse of special round */
KEYXOR64 (x0,x1,x2,x3, ctx->keys[32]);
SBOX7_INVERSE (x0,x1,x2,x3, y0,y1,y2,y3);
KEYXOR64 (y0,y1,y2,y3, ctx->keys[31]);

k = 24;
goto start64;
while (k > 0)
{
    k -= 8;
    ROUND64_INVERSE (7, ctx->keys[k+7], x0,x1,x2,x3, y0,y1,y2,y3);
start64:
    ROUND64_INVERSE (6, ctx->keys[k+6], y0,y1,y2,y3, x0,x1,x2,x3);
    ROUND64_INVERSE (5, ctx->keys[k+5], x0,x1,x2,x3, y0,y1,y2,y3);
    ROUND64_INVERSE (4, ctx->keys[k+4], y0,y1,y2,y3, x0,x1,x2,x3);
    ROUND64_INVERSE (3, ctx->keys[k+3], x0,x1,x2,x3, y0,y1,y2,y3);
    ROUND64_INVERSE (2, ctx->keys[k+2], y0,y1,y2,y3, x0,x1,x2,x3);
    ROUND64_INVERSE (1, ctx->keys[k+1], x0,x1,x2,x3, y0,y1,y2,y3);
    ROUND64_INVERSE (0, ctx->keys[k], y0,y1,y2,y3, x0,x1,x2,x3);
}

LE_WRITE_UINT32 (dst + 16, x0);
LE_WRITE_UINT32 (dst + 20, x1);
LE_WRITE_UINT32 (dst + 24, x2);
LE_WRITE_UINT32 (dst + 28, x3);
x0 >>= 32; LE_WRITE_UINT32 (dst, x0);
x1 >>= 32; LE_WRITE_UINT32 (dst + 4, x1);
x2 >>= 32; LE_WRITE_UINT32 (dst + 8, x2);
x3 >>= 32; LE_WRITE_UINT32 (dst + 12, x3);
}
#endif /* HAVE_NATIVE_64_BIT */
}
/* serpent-set-key.c

```

The serpent block cipher.

For more details on this algorithm, see the Serpent website at
<http://www.cl.cam.ac.uk/~rja14/serpent.html>

Copyright (C) 2011, 2014 Niels Möller
 Copyright (C) 2010, 2011 Simon Josefsson
 Copyright (C) 2003, 2004, 2005 Free Software Foundation, Inc.

This file is part of GNU Nettle.

GNU Nettle is free software: you can redistribute it and/or modify it under the terms of either:

- * the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

or

- * the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

or both in parallel, as here.

GNU Nettle is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received copies of the GNU General Public License and the GNU Lesser General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

*/

```
/* This file is derived from cipher/serpent.c in Libgcrypt v1.4.6.
   The adaption to Nettle was made by Simon Josefsson on 2010-12-07
   with final touches on 2011-05-30. Changes include replacing
   libgcrypt with nettle in the license template, renaming
   serpent_context to serpent_ctx, renaming u32 to uint32_t, removing
   libgcrypt stubs and selftests, modifying entry function prototypes,
   using FOR_BLOCKS to iterate through data in encrypt/decrypt, using
   LE_READ_UINT32 and LE_WRITE_UINT32 to access data in
   encrypt/decrypt, and running indent on the code. */
```

```
#if HAVE_CONFIG_H
#include "config.h"
#endif
```

```
#include <assert.h>
#include <limits.h>
```

```
#include "serpent.h"
```

```
#include "macros.h"
#include "serpent-internal.h"
```

```
/* Magic number, used during generating of the subkeys. */
#define PHI 0x9E3779B9
```

```
/* These are the S-Boxes of Serpent. They are copied from Serpents
   reference implementation (the optimized one, contained in
   `floppy2') and are therefore:
```

Copyright (C) 1998 Ross Anderson, Eli Biham, Lars Knudsen.

To quote the Serpent homepage
(<http://www.cl.cam.ac.uk/~rja14/serpent.html>):

"Serpent is now completely in the public domain, and we impose no restrictions on its use. This was announced on the 21st August at the First AES Candidate Conference. The optimised implementations in the submission package are now under the GNU PUBLIC LICENSE (GPL), although some comments in the code still say otherwise. You are welcome to use Serpent for any application." */

/* FIXME: Except when used within the key schedule, the inputs are not used after the substitution, and hence we could allow them to be destroyed. Can this freedom be used to optimize the sboxes? */

```
#define SBOX0(type, a, b, c, d, w, x, y, z)\
do {\
    type t02, t03, t05, t06, t07, t08, t09; \
    type t11, t12, t13, t14, t15, t17, t01; \
    t01 = b ^ c ; \
    t02 = a | d ; \
    t03 = a ^ b ; \
    z = t02 ^ t01; \
    t05 = c | z ; \
    t06 = a ^ d ; \
    t07 = b | c ; \
    t08 = d & t05; \
    t09 = t03 & t07; \
    y = t09 ^ t08; \
    t11 = t09 & y ; \
    t12 = c ^ d ; \
    t13 = t07 ^ t11; \
    t14 = b & t06; \
    t15 = t06 ^ t13; \
    w = ~ t15; \
    t17 = w ^ t14; \
    x = t12 ^ t17; \
} while (0)
```

```
#define SBOX1(type, a, b, c, d, w, x, y, z)\
do {\
    type t02, t03, t04, t05, t06, t07, t08; \
    type t10, t11, t12, t13, t16, t17, t01; \
    t01 = a | d ; \
    t02 = c ^ d ; \
    t03 = ~ b ; \
    t04 = a ^ c ; \
    t05 = a | t03; \
    t06 = d & t04; \
    t07 = t01 & t02; \
    t08 = b | t06; \
    y = t02 ^ t05; \
    t10 = t07 ^ t08; \
    t11 = t01 ^ t10; \
```

```

t12 = y ^ t11; \
t13 = b & d ; \
z = ~ t10; \
x = t13 ^ t12; \
t16 = t10 | x ; \
t17 = t05 & t16; \
w = c ^ t17; \
} while (0)

```

```

#define SBOX2(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t05, t06, t07, t08; \
    type t09, t10, t12, t13, t14, t01; \
    t01 = a | c ; \
    t02 = a ^ b ; \
    t03 = d ^ t01; \
    w = t02 ^ t03; \
    t05 = c ^ w ; \
    t06 = b ^ t05; \
    t07 = b | t05; \
    t08 = t01 & t06; \
    t09 = t03 ^ t07; \
    t10 = t02 | t09; \
    x = t10 ^ t08; \
    t12 = a | d ; \
    t13 = t09 ^ x ; \
    t14 = b ^ t13; \
    z = ~ t09; \
    y = t12 ^ t14; \
} while (0)

```

```

#define SBOX3(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t04, t05, t06, t07, t08; \
    type t09, t10, t11, t13, t14, t15, t01; \
    t01 = a ^ c ; \
    t02 = a | d ; \
    t03 = a & d ; \
    t04 = t01 & t02; \
    t05 = b | t03; \
    t06 = a & b ; \
    t07 = d ^ t04; \
    t08 = c | t06; \
    t09 = b ^ t07; \
    t10 = d & t05; \
    t11 = t02 ^ t10; \
    z = t08 ^ t09; \
    t13 = d | z ; \
    t14 = a | t07; \
    t15 = b & t13; \
    y = t08 ^ t11; \
    w = t14 ^ t15; \
    x = t05 ^ t04; \
} while (0)

```



```
#define SBOX4(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t04, t05, t06, t08, t09; \
    type t10, t11, t12, t13, t14, t15, t16, t01; \
    t01 = a | b ; \
    t02 = b | c ; \
    t03 = a ^ t02; \
    t04 = b ^ d ; \
    t05 = d | t03; \
    t06 = d & t01; \
    z = t03 ^ t06; \
    t08 = z & t04; \
    t09 = t04 & t05; \
    t10 = c ^ t06; \
    t11 = b & c ; \
    t12 = t04 ^ t08; \
    t13 = t11 | t03; \
    t14 = t10 ^ t09; \
    t15 = a & t05; \
    t16 = t11 | t12; \
    y = t13 ^ t08; \
    x = t15 ^ t16; \
    w = ~ t14; \
} while (0)
```

```
#define SBOX5(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t04, t05, t07, t08, t09; \
    type t10, t11, t12, t13, t14, t01; \
    t01 = b ^ d ; \
    t02 = b | d ; \
    t03 = a & t01; \
    t04 = c ^ t02; \
    t05 = t03 ^ t04; \
    w = ~ t05; \
    t07 = a ^ t01; \
    t08 = d | w ; \
    t09 = b | t05; \
    t10 = d ^ t08; \
    t11 = b | t07; \
    t12 = t03 | w ; \
    t13 = t07 | t10; \
    t14 = t01 ^ t11; \
    y = t09 ^ t13; \
    x = t07 ^ t08; \
    z = t12 ^ t14; \
} while (0)
```

```
#define SBOX6(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t04, t05, t07, t08, t09, t10; \
    type t11, t12, t13, t15, t17, t18, t01; \
    t01 = a & d ; \
    t02 = b ^ c ; \

```

```

t03 = a ^ d ; \
t04 = t01 ^ t02; \
t05 = b | c ; \
x = ~ t04; \
t07 = t03 & t05; \
t08 = b & x ; \
t09 = a | c ; \
t10 = t07 ^ t08; \
t11 = b | d ; \
t12 = c ^ t11; \
t13 = t09 ^ t10; \
y = ~ t13; \
t15 = x & t03; \
z = t12 ^ t07; \
t17 = a ^ b ; \
t18 = y ^ t15; \
w = t17 ^ t18; \
} while (0)

```

```

#define SBOX7(type, a, b, c, d, w, x, y, z) \
do { \
    type t02, t03, t04, t05, t06, t08, t09, t10; \
    type t11, t13, t14, t15, t16, t17, t01; \
    t01 = a & c ; \
    t02 = ~ d ; \
    t03 = a & t02; \
    t04 = b | t01; \
    t05 = a & b ; \
    t06 = c ^ t04; \
    z = t03 ^ t06; \
    t08 = c | z ; \
    t09 = d | t05; \
    t10 = a ^ t08; \
    t11 = t04 & z ; \
    x = t09 ^ t10; \
    t13 = b ^ x ; \
    t14 = t01 ^ x ; \
    t15 = c ^ t05; \
    t16 = t11 | t13; \
    t17 = t02 | t14; \
    w = t15 ^ t17; \
    y = a ^ t16; \
} while (0)

```

```

/* Key schedule */
/* Note: Increments k */
#define KS_RECURRENCE(w, i, k)\
do {\
    uint32_t _wn = (w)[(i)] ^ (w)[((i)+3)&7] ^ w[((i)+5)&7] \
    ^ w[((i)+7)&7] ^ PHI ^ (k)++; \
    ((w)[(i)] = ROTL32(11, _wn)); \
} while (0)

```

```

/* Note: Increments k four times and keys once */
#define KS(keys, s, w, i, k)\

```

```

do {\
    KS_RECURRENCE(w, (i), (k));\
    KS_RECURRENCE(w, (i)+1, (k));\
    KS_RECURRENCE(w, (i)+2, (k));\
    KS_RECURRENCE(w, (i)+3, (k));\
    SBOX##s(uint32_t, w[(i)],w[(i)+1],w[(i)+2],w[(i)+3],\
    (*keys)[0],(*keys)[1],(*keys)[2],(*keys)[3]);\
    (keys)++;\
} while (0)

/* Pad user key and convert to an array of 8 uint32_t. */
static void
serpent_key_pad (const uint8_t *key, unsigned int key_length,
uint32_t *w)
{
    unsigned int i;

    assert (key_length <= SERPENT_MAX_KEY_SIZE);

    for (i = 0; key_length >= 4; key_length -=4, key += 4)
        w[i++] = LE_READ_UINT32(key);

    if (i < 8)
    {
        /* Key must be padded according to the Serpent specification.
        "aabbcc" -> "aabbcc0100...00" -> 0x01ccbbaa. */
        uint32_t pad = 0x01;

        while (key_length > 0)
            pad = pad << 8 | key[--key_length];

        w[i++] = pad;

        while (i < 8)
            w[i++] = 0;
    }
}

/* Initialize CONTEXT with the key KEY of LENGTH bytes. */
void
serpent_set_key (struct serpent_ctx *ctx,
size_t length, const uint8_t *key)
{
    uint32_t w[8];
    uint32_t (*keys)[4];
    unsigned k;

    serpent_key_pad (key, length, w);

    /* Derive the 33 subkeys from KEY and store them in SUBKEYS. We do
    the recurrence in the key schedule using W as a circular buffer
    of just 8 uint32_t. */

    /* FIXME: Would be better to invoke SBOX with scalar variables as

```

arguments, no arrays. To do that, unpack w into separate variables, use temporary variables as the SBOX destination. */

```
keys = ctx->keys;
k = 0;
for (;;)
{
    KS(keys, 3, w, 0, k);
    if (k == 132)
break;
    KS(keys, 2, w, 4, k);
    KS(keys, 1, w, 0, k);
    KS(keys, 0, w, 4, k);
    KS(keys, 7, w, 0, k);
    KS(keys, 6, w, 4, k);
    KS(keys, 5, w, 0, k);
    KS(keys, 4, w, 4, k);
}
assert (keys == ctx->keys + 33);
}

void
serpent128_set_key (struct serpent_ctx *ctx, const uint8_t *key)
{
    serpent_set_key (ctx, SERPENT128_KEY_SIZE, key);
}

void
serpent192_set_key (struct serpent_ctx *ctx, const uint8_t *key)
{
    serpent_set_key (ctx, SERPENT192_KEY_SIZE, key);
}

void
serpent256_set_key (struct serpent_ctx *ctx, const uint8_t *key)
{
    serpent_set_key (ctx, SERPENT256_KEY_SIZE, key);
}
```

1.100 nfs-utils

1.100.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.101 ntp

1.101.1 Available under license

This file is automatically generated from `html/copyright.html`
Copyright Notice

jpg "Clone me," says Dolly sheepishly.

Last update: 4-Feb-2020 23:47 UTC

The following copyright notice applies to all files collectively called the Network Time Protocol Version 4 Distribution. Unless specifically declared otherwise in an individual file, this entire notice applies as if the text was explicitly included in the file.

```
*****
*                                     *
* Copyright (c) University of Delaware 1992-2015      *
*                                     *
* Permission to use, copy, modify, and distribute this software and *
* its documentation for any purpose with or without fee is hereby *
* granted, provided that the above copyright notice appears in all *
* copies and that both the copyright notice and this permission *
* notice appear in supporting documentation, and that the name *
* University of Delaware not be used in advertising or publicity *
* pertaining to distribution of the software without specific, *
* written prior permission. The University of Delaware makes no *
* representations about the suitability this software for any *
* purpose. It is provided "as is" without express or implied *
* warranty.                                           *
*                                     *
*****
```

Content starting in 2011 from Harlan Stenn, Danny Mayer, and Martin Burnicki is:

```
*****
*                                     *
* Copyright (c) Network Time Foundation 2011-2020      *
*                                     *
* All Rights Reserved                                *
*                                     *
* Redistribution and use in source and binary forms, with or without *
* modification, are permitted provided that the following conditions *
* are met:                                           *
* 1. Redistributions of source code must retain the above copyright *
* notice, this list of conditions and the following disclaimer. *
* 2. Redistributions in binary form must reproduce the above *
* copyright notice, this list of conditions and the following *
* disclaimer in the documentation and/or other materials provided *
* with the distribution.                             *
*                                     *
*****
```

```

* THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS" AND ANY EXPRESS *
* OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED *
* WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE *
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE *
* LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR *
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT *
* OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR *
* BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF *
* LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT *
* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE *
* USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH *
* DAMAGE.

```

```

*****

```

The following individuals contributed in part to the Network Time Protocol Distribution Version 4 and are acknowledged as authors of this work.

1. [1]Takao Abe <takao_abe@xurb.jp> Clock driver for JJY receivers
2. [2]Mark Andrews <mark_andrews@isc.org> Leitch atomic clock controller
3. [3]Bernd Altmeier <altmeier@atlsoft.de> hopf Elektronik serial line and PCI-bus devices
4. [4]Viraj Bais <vbais@mailman1.intel.com> and [5]Clayton Kirkwood <kirkwood@striderfin.intel.com> port to WindowsNT 3.5
5. [6]Michael Barone <michael.barone@lmco.com> GPSVME fixes
6. [7]Karl Berry <karl@owl.HQ.ileaf.com> syslog to file option
7. [8]Greg Brackley <greg.brackley@bigfoot.com> Major rework of WINNT port. Clean up recvbuf and iosignal code into separate modules.
8. [9]Marc Brett <Marc.Brett@westgeo.com> Magnavox GPS clock driver
9. [10]Piete Brooks <Piete.Brooks@cl.cam.ac.uk> MSF clock driver, Trimble PARSE support
10. [11]Nelson B Bolyard <nelson@bolyard.me> update and complete broadcast and crypto features in snpt
11. [12]Jean-Francois Boudreault <Jean-Francois.Boudreault@viagenie.qc.ca> IPv6 support
12. [13]Reg Clemens <reg@dwf.com> Oncore driver (Current maintainer)
13. [14]Steve Clift <clift@ml.csiro.au> OMEGA clock driver
14. [15]Casey Crellin <casey@csc.co.za> vxWorks (Tornado) port and help with target configuration
15. [16]Sven Dietrich <sven_dietrich@trimble.com> Palisade reference clock driver, NT adj. residuals, integrated Greg's Winnt port.
16. [17]John A. Dundas III <dundas@salt.jpl.nasa.gov> Apple A/UX port
17. [18]Torsten Duwe <duwe@immd4.informatik.uni-erlangen.de> Linux port
18. [19]Dennis Ferguson <dennis@mrbill.canet.ca> foundation code for NTP Version 2 as specified in RFC-1119
19. [20]John Hay <jhay@icomtek.csir.co.za> IPv6 support and testing
20. [21]Dave Hart <davehart@davehart.com> General maintenance, Windows port interpolation rewrite
21. [22]Claas Hilbrecht <neoclock4x@linum.com> NeoClock4X clock driver
22. [23]Glenn Hollinger <glenn@herald.usask.ca> GOES clock driver
23. [24]Mike Iglesias <iglesias@uci.edu> DEC Alpha port
24. [25]Jim Jagielski <jim@jagubox.gsfc.nasa.gov> A/UX port
25. [26]Jeff Johnson <jbj@chatham.usdesign.com> massive prototyping overhaul

26. [27]Hans Lambermont <Hans.Lambermont@nl.origin-it.com> or
[28]<H.Lambermont@chello.nl> ntpswEEP
27. [29]Poul-Henning Kamp <phk@FreeBSD.ORG> Oncore driver (Original author)
28. [30]Frank Kardel [31]<kardel (at) ntp (dot) org> PARSE <GENERIC> (driver 14 reference clocks), STREAMS modules for PARSE, support scripts, syslog cleanup, dynamic interface handling
29. [32]Johannes Maximilian Kuehn <kuehn@ntp.org> Rewrote snTP to comply with NTPv4 specification, ntpq saveconfig
30. [33]William L. Jones <jones@hermes.chpc.utexas.edu> RS/6000 AIX modifications, HP-UX modifications
31. [34]Dave Katz <dkatz@cisco.com> RS/6000 AIX port
32. [35]Craig Leres <leres@ee.lbl.gov> 4.4BSD port, ppsClock, Magnavox GPS clock driver
33. [36]George Lindholm <lindholm@ucs.ubc.ca> SunOS 5.1 port
34. [37]Louis A. Mamakos <louie@ni.umd.edu> MD5-based authentication
35. [38]Lars H. Mathiesen <thorinn@diku.dk> adaptation of foundation code for Version 3 as specified in RFC-1305
36. [39]Danny Mayer <mayer@ntp.org> Network I/O, Windows Port, Code Maintenance
37. [40]David L. Mills <mills@udel.edu> Version 4 foundation, precision kernel; clock drivers: 1, 3, 4, 6, 7, 11, 13, 18, 19, 22, 36
38. [41]Wolfgang Moeller <moeller@gwdgv1.dnet.gwdg.de> VMS port
39. [42]Jeffrey Mogul <mogul@pa.dec.com> ntptrace utility
40. [43]Tom Moore <tmoore@fielvel.daytonoh.ncr.com> i386 svr4 port
41. [44]Kamal A Mostafa <kamal@whence.com> SCO OpenServer port
42. [45]Derek Mulcahy <derek@toybox.demon.co.uk> and [46]Damon Hart-Davis <d@hd.org> ARCRON MSF clock driver
43. [47]Rob Neal <neal@ntp.org> Bancomm refclock and config/parse code maintenance
44. [48]Rainer Pruy <Rainer.Pruy@informatik.uni-erlangen.de> monitoring/trap scripts, statistics file handling
45. [49]Dirce Richards <dirce@zk3.dec.com> Digital UNIX V4.0 port
46. [50]Wilfredo Sánchez <wsanchez@apple.com> added support for NetInfo
47. [51]Nick Sayer <mrapple@quack.kfu.com> SunOS streams modules
48. [52]Jack Sasportas <jack@innovativeinternet.com> Saved a Lot of space on the stuff in the html/pic/ subdirectory
49. [53]Ray Schnitzler <schnitz@unipress.com> Unixware1 port
50. [54]Michael Shields <shields@tembel.org> USNO clock driver
51. [55]Jeff Steinman <jss@pebbles.jpl.nasa.gov> Datum PTS clock driver
52. [56]Harlan Stenn <harlan@pfcs.com> GNU automake/autoconfigure makeover, various other bits (see the ChangeLog)
53. [57]Kenneth Stone <ken@sdd.hp.com> HP-UX port
54. [58]Ajit Thyagarajan <ajit@ee.udel.edu> IP multicast/anycast support
55. [59]Tomoaki TSURUOKA <tsuruoka@nc.fukuoka-u.ac.jp> TRAK clock driver
56. [60]Brian Utterback <brian.utterback@oracle.com> General codebase, Solaris issues
57. [61]Loganaden Velvindron <loganaden@gmail.com> Sandboxing (libseccomp) support
58. [62]Paul A Vixie <vixie@vix.com> TrueTime GPS driver, generic TrueTime clock driver
59. [63]Ulrich Windl <Ulrich.Windl@rz.uni-regensburg.de> corrected and validated HTML documents according to the HTML DTD

References

1. mailto:%20takao_abe@xurb.jp
2. mailto:%20mark_andrews@isc.org
3. <mailto:%20altmeier@atlsoft.de>
4. <mailto:%20vbais@mailman1.intel.co>
5. <mailto:%20kirkwood@striderfm.intel.com>
6. <mailto:%20michael.barone@lmco.com>
7. <mailto:%20karl@owl.HQ.ileaf.com>
8. <mailto:%20greg.brackley@bigfoot.com>
9. <mailto:%20Marc.Brett@westgeo.com>
10. <mailto:%20Piete.Brooks@cl.cam.ac.uk>
11. <mailto:%20nelson@bolyard.me>
12. <mailto:%20Jean-Francois.Boudreault@viagenie.qc.ca>
13. <mailto:%20reg@dwf.com>
14. <mailto:%20clift@ml.csiro.au>
15. <mailto:%20casey@csc.co.za>
16. mailto:%20Sven_Dietrich@trimble.COM
17. <mailto:%20dundas@salt.jpl.nasa.gov>
18. <mailto:%20duwe@immd4.informatik.uni-erlangen.de>
19. <mailto:%20dennis@mrbill.canet.ca>
20. <mailto:%20jhay@icomtek.csir.co.za>
21. <mailto:%20davehart@davehart.com>
22. <mailto:%20neoclock4x@linum.com>
23. <mailto:%20glenn@herald.usask.ca>
24. <mailto:%20iglesias@uci.edu>
25. <mailto:%20jagubox.gsfc.nasa.gov>
26. <mailto:%20bjb@chatham.usdesign.com>
27. <mailto:%20Hans.Lambermont@nl.origin-it.com>
28. <mailto:H.Lambermont@chello.nl>
29. <mailto:%20phk@FreeBSD.ORG>
30. <http://www4.informatik.uni-erlangen.de/%7ekardel>
31. <mailto:%20kardel%20%28at%29%20ntp%20%28dot%29%20org>
32. <mailto:kuehn@ntp.org>
33. <mailto:%20jones@hermes.chpc.utexas.edu>
34. <mailto:%20dkatz@cisco.com>
35. <mailto:%20leres@ee.lbl.gov>
36. <mailto:%20lindholm@ucs.ubc.ca>
37. <mailto:%20louie@ni.umd.edu>
38. <mailto:%20thorinn@diku.dk>
39. <mailto:%20mayer@ntp.org>
40. <mailto:%20mills@udel.edu>
41. <mailto:%20moeller@gwdgv1.dnet.gwdg.de>
42. <mailto:%20mogul@pa.dec.com>
43. <mailto:%20tmoore@fivel.daytonoh.ncr.com>
44. <mailto:%20kamal@whence.com>
45. <mailto:%20derek@toybox.demon.co.uk>
46. <mailto:%20d@hd.org>
47. <mailto:%20neal@ntp.org>
48. <mailto:%20Rainer.Pruy@informatik.uni-erlangen.de>
49. <mailto:%20dirce@zk3.dec.com>
50. <mailto:%20wsanchez@apple.com>
51. <mailto:%20mrapple@quack.kfu.com>
52. <mailto:%20jack@innovativeinternet.com>

- 53. <mailto:%20schnittz@unipress.com>
- 54. <mailto:%20shields@tembel.org>
- 55. <mailto:%20pebbles.jpl.nasa.gov>
- 56. <mailto:%20harlan@pfcs.com>
- 57. <mailto:%20ken@sdd.hp.com>
- 58. <mailto:%20ajit@ee.udel.edu>
- 59. <mailto:%20tsuruoka@nc.fukuoka-u.ac.jp>
- 60. <mailto:%20brian.utterback@oracle.com>
- 61. <mailto:%20loganaden@gmail.com>
- 62. <mailto:%20vixie@vix.com>
- 63. <mailto:%20Ulrich.Windl@rz.uni-regensburg.de>

1.102 openais

1.102.1 Available under license

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.103 openssh

1.103.1 Available under license

This file is part of the OpenSSH software.

The licences which components of this software fall under are as follows. First, we will summarize and say that all components are under a BSD licence, or a licence more free than that.

OpenSSH contains no GPL code.

1)

- * Copyright (c) 1995 Tatu Ylonen <ylo@cs.hut.fi>, Espoo, Finland
- * All rights reserved
- *
- * As far as I am concerned, the code I have written for this software
- * can be used freely for any purpose. Any derived versions of this
- * software must be clearly marked as such, and if the derived work is
- * incompatible with the protocol description in the RFC file, it must be
- * called by a name other than "ssh" or "Secure Shell".

[Tatu continues]

- * However, I am not implying to give any licenses to any patents or
- * copyrights held by third parties, and the software includes parts that
- * are not under my direct control. As far as I know, all included
- * source code is used in accordance with the relevant license agreements
- * and can be used freely for any purpose (the GNU license being the most
- * restrictive); see below for details.

[However, none of that term is relevant at this point in time. All of these restrictively licenced software components which he talks about have been removed from OpenSSH, i.e.,

- RSA is no longer included, found in the OpenSSL library
- IDEA is no longer included, its use is deprecated
- DES is now external, in the OpenSSL library
- GMP is no longer used, and instead we call BN code from OpenSSL
- Zlib is now external, in a library
- The make-ssh-known-hosts script is no longer included
- TSS has been removed
- MD5 is now external, in the OpenSSL library
- RC4 support has been replaced with ARC4 support from OpenSSL
- Blowfish is now external, in the OpenSSL library

[The licence continues]

Note that any information and cryptographic algorithms used in this software are publicly available on the Internet and at any major bookstore, scientific library, and patent office worldwide. More information can be found e.g. at "<http://www.cs.hut.fi/crypto>".

The legal status of this program is some combination of all these permissions and restrictions. Use only at your own responsibility. You will be responsible for any legal consequences yourself; I am not making any claims whether possessing or using this is legal or not in your country, and I am not taking any responsibility on your behalf.

NO WARRANTY

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

- 3)
ssh-keyscan was contributed by David Mazieres under a BSD-style license.

- * Copyright 1995, 1996 by David Mazieres <dm@lcs.mit.edu>.
- *
- * Modification and redistribution in source and binary forms is
- * permitted provided that due credit is given to the author and the
- * OpenBSD project by leaving this copyright notice intact.

- 4)
The Rijndael implementation by Vincent Rijmen, Antoon Bosselaers and Paulo Barreto is in the public domain and distributed with the following license:

- * @version 3.0 (December 2000)
- *
- * Optimised ANSI C code for the Rijndael cipher (now AES)
- *
- * @author Vincent Rijmen <vincent.rijmen@esat.kuleuven.ac.be>
- * @author Antoon Bosselaers <antoon.bosselaers@esat.kuleuven.ac.be>
- * @author Paulo Barreto <paulo.barreto@terra.com.br>
- *
- * This code is hereby placed in the public domain.
- *

* THIS SOFTWARE IS PROVIDED BY THE AUTHORS "AS IS" AND ANY EXPRESS
 * OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
 * BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
 * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
 * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5)

One component of the ssh source code is under a 3-clause BSD license,
 held by the University of California, since we pulled these parts from
 original Berkeley code.

* Copyright (c) 1983, 1990, 1992, 1993, 1995
 * The Regents of the University of California. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the University nor the names of its contributors
 * may be used to endorse or promote products derived from this software
 * without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.

6)

Remaining components of the software are provided under a standard
 2-term BSD licence with the following names as copyright holders:

Markus Friedl
 Theo de Raadt
 Niels Provos
 Dug Song
 Aaron Campbell
 Damien Miller
 Kevin Steves

Daniel Kouril
Wesley Griffin
Per Allansson
Nils Nordman
Simon Wilkinson

Portable OpenSSH additionally includes code from the following copyright holders, also under the 2-term BSD license:

Ben Lindstrom
Tim Rice
Andre Lucas
Chris Adams
Corinna Vinschen
Cray Inc.
Denis Parker
Gert Doering
Jakob Schlyter
Jason Downs
Juha Yrjölä
Michael Stone
Networks Associates Technology, Inc.
Solar Designer
Todd C. Miller
Wayne Schroeder
William Jones
Darren Tucker
Sun Microsystems
The SCO Group
Daniel Walsh
Red Hat, Inc
Simon Vallet / Genoscope

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:

- * 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
- * 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.

*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR
* IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
* IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
* INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
* DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
* THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
* THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8) Portable OpenSSH contains the following additional licenses:

a) md5crypt.c, md5crypt.h

- * "THE BEER-WARE LICENSE" (Revision 42):
- * <phk@login.dknet.dk> wrote this file. As long as you retain this
- * notice you can do whatever you want with this stuff. If we meet
- * some day, and you think this stuff is worth it, you can buy me a
- * beer in return. Poul-Henning Kamp

b) snprintf replacement

- * Copyright Patrick Powell 1995
- * This code is based on code written by Patrick Powell
- * (papowell@astart.com) It may be used for any purpose as long as this
- * notice remains intact on all source code distributions

c) Compatibility code (openbsd-compat)

Apart from the previously mentioned licenses, various pieces of code in the openbsd-compat/ subdirectory are licensed as follows:

Some code is licensed under a 3-term BSD license, to the following copyright holders:

Todd C. Miller
 Theo de Raadt
 Damien Miller
 Eric P. Allman
 The Regents of the University of California
 Constantin S. Svintsoff

- * Redistribution and use in source and binary forms, with or without
- * modification, are permitted provided that the following conditions
- * are met:
- * 1. Redistributions of source code must retain the above copyright
- * notice, this list of conditions and the following disclaimer.
- * 2. Redistributions in binary form must reproduce the above copyright
- * notice, this list of conditions and the following disclaimer in the
- * documentation and/or other materials provided with the distribution.
- * 3. Neither the name of the University nor the names of its contributors
- * may be used to endorse or promote products derived from this software
- * without specific prior written permission.
- *
- * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
- * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
- * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
- * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
- * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
- * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
- * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
- * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
- * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
- * SUCH DAMAGE.

Some code is licensed under an ISC-style license, to the following copyright holders:

Internet Software Consortium.

Todd C. Miller

Reyk Floeter

Chad Mynhier

* Permission to use, copy, modify, and distribute this software for any
* purpose with or without fee is hereby granted, provided that the above
* copyright notice and this permission notice appear in all copies.
*

* THE SOFTWARE IS PROVIDED "AS IS" AND TODD C. MILLER DISCLAIMS ALL
* WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL TODD C. MILLER BE LIABLE
* FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
* WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION
* OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN
* CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Some code is licensed under a MIT-style license to the following copyright holders:

Free Software Foundation, Inc.

* Permission is hereby granted, free of charge, to any person obtaining a *
* copy of this software and associated documentation files (the *
* "Software"), to deal in the Software without restriction, including *
* without limitation the rights to use, copy, modify, merge, publish, *
* distribute, distribute with modifications, sublicense, and/or sell *
* copies of the Software, and to permit persons to whom the Software is *
* furnished to do so, subject to the following conditions: *

* The above copyright notice and this permission notice shall be included *
* in all copies or substantial portions of the Software. *

* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS *
* OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF *
* MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. *
* IN NO EVENT SHALL THE ABOVE COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, *
* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR *
* OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR *
* THE USE OR OTHER DEALINGS IN THE SOFTWARE. *

* Except as contained in this notice, the name(s) of the above copyright *
* holders shall not be used in advertising or otherwise to promote the *
* sale, use or other dealings in this Software without prior written *
* authorization. *

*****/

The Blowfish cipher implementation is licensed by Niels Provis under a 4-clause BSD license:

* Blowfish - a fast block cipher designed by Bruce Schneier

```

*
* Copyright 1997 Niels Provos <provos@physnet.uni-hamburg.de>
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
*   must display the following acknowledgement:
*   This product includes software developed by Niels Provos.
* 4. The name of the author may not be used to endorse or promote products
*   derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR
* IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
* IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
* INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
* DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
* THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
* THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```

Some replacement code is licensed by the NetBSD foundation under a 2-clause BSD license:

```

* Copyright (c) 2001 The NetBSD Foundation, Inc.
* All rights reserved.
*
* This code is derived from software contributed to The NetBSD Foundation
* by Todd Vierling.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
*
* THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS
* ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
* TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS
* BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
* SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN

```


- * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
- * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
- * POSSIBILITY OF SUCH DAMAGE.

\$OpenBSD: LICENCE,v 1.20 2017/04/30 23:26:16 djm Exp \$

1.104 openssl

1.104.1 Available under license

LICENSE ISSUES

The OpenSSL toolkit stays under a double license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts.

OpenSSL License

```
/* =====
 * Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in
 * the documentation and/or other materials provided with the
 * distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 * software must display the following acknowledgment:
 * "This product includes software developed by the OpenSSL Project
 * for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
 *
 * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
 * endorse or promote products derived from this software without
 * prior written permission. For written permission, please contact
 * openssl-core@openssl.org.
 *
 * 5. Products derived from this software may not be called "OpenSSL"
 * nor may "OpenSSL" appear in their names without prior written
 * permission of the OpenSSL Project.
 *
 * 6. Redistributions of any form whatsoever must retain the following
 * acknowledgment:
 * "This product includes software developed by the OpenSSL Project
 * for use in the OpenSSL Toolkit (http://www.openssl.org/)"
 *
 * THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
 * EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
```

* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
 * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
 * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGE.

* =====

*
 * This product includes cryptographic software written by Eric Young
 * (eay@cryptsoft.com). This product includes software written by Tim
 * Hudson (tjh@cryptsoft.com).
 *
 */

Original SSLeay License

```
/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
 * All rights reserved.
 *
 * This package is an SSL implementation written
 * by Eric Young (eay@cryptsoft.com).
 * The implementation was written so as to conform with Netscapes SSL.
 *
 * This library is free for commercial and non-commercial use as long as
 * the following conditions are aheared to. The following conditions
 * apply to all code found in this distribution, be it the RC4, RSA,
 * lhash, DES, etc., code; not just the SSL code. The SSL documentation
 * included with this distribution is covered by the same copyright terms
 * except that the holder is Tim Hudson (tjh@cryptsoft.com).
 *
 * Copyright remains Eric Young's, and as such any Copyright notices in
 * the code are not to be removed.
 * If this package is used in a product, Eric Young should be given attribution
 * as the author of the parts of the library used.
 * This can be in the form of a textual message at program startup or
 * in documentation (online or textual) provided with the package.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 *    "This product includes cryptographic software written by
 *     Eric Young (eay@cryptsoft.com)"
 * The word 'cryptographic' can be left out if the rouines from the library
```

```
*  being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
*    the apps directory (application code) you must include an acknowledgement:
*    "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed.  i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/
```

1.105 opkg-utils

1.105.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

```
#!/usr/bin/env python
# Copyright (C) 2001 Alexander S. Guy <a7r@andern.org>
#      Andern Research Labs
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2, or (at your option)
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place - Suite 330,
# Boston, MA 02111-1307, USA. */
#
# Copyright 2001, Russell Nelson <opkg.py@russnelson.com>
# Added reading in of packages.
# Added missing package information fields.
# Changed render_control() to __repr__().
#
# Current Issues:
# The API doesn't validate package information fields. It should be
# throwing exceptions in the right places.
# Executions of tar could silently fail.
# Executions of tar *do* fail, and loudly, because you have to specify a full filename,
# and tar complains if any files are missing, and the opkg spec doesn't require
# people to say "./control.tar.gz" or "./control" when they package files.
# It would be much better to require ./control or disallow ./control (either)
# rather than letting people pick. Some freedoms aren't worth their cost.
from __future__ import absolute_import
```

```

from __future__ import print_function

import tempfile
import os
import sys
import glob
import hashlib
import re
import subprocess
from stat import ST_SIZE
import arfile
import tarfile
import textwrap

class Version(object):
    """A class for holding parsed package version information."""
    def __init__(self, epoch, version):
        self.epoch = epoch
        self.version = version

    def _versioncompare(self, selfversion, refversion):
        if not selfversion: selfversion = ""
        if not refversion: refversion = ""
        while 1:
            ## first look for non-numeric version component
            selfm = re.match('^0-9]*)(.*)', selfversion)
            #print(('selfm', selfm.groups()))
            (selfalpha, selfversion) = selfm.groups()
            refm = re.match('^0-9]*)(.*)', refversion)
            #print(('refm', refm.groups()))
            (refalpha, refversion) = refm.groups()
            if (selfalpha > refalpha):
                return 1
            elif (selfalpha < refalpha):
                return -1
            ## now look for numeric version component
            (selfnum, selfversion) = re.match('([0-9]*)(.*)', selfversion).groups()
            (refnum, refversion) = re.match('([0-9]*)(.*)', refversion).groups()
            #print(('selfnum', selfnum, selfversion))
            #print(('refnum', refnum, refversion))
            if (selfnum != ""):
                selfnum = int(selfnum)
            else:
                selfnum = -1
            if (refnum != ""):
                refnum = int(refnum)
            else:
                refnum = -1
            if (selfnum > refnum):
                return 1
            elif (selfnum < refnum):
                return -1
            if selfversion == "" and refversion == "":
                return 0

```

```

def compare(self, ref):
    if (self.epoch > ref.epoch):
        return 1
    elif (self.epoch < ref.epoch):
        return -1
    else:
        self_ver_comps = re.match(r"(.+?)(-r.+)?$", self.version)
        ref_ver_comps = re.match(r"(.+?)(-r.+)?$", ref.version)
        #print((self_ver_comps.group(1), self_ver_comps.group(2)))
        #print((ref_ver_comps.group(1), ref_ver_comps.group(2)))
        r = self._versioncompare(self_ver_comps.group(1), ref_ver_comps.group(1))
        if r == 0:
            r = self._versioncompare(self_ver_comps.group(2), ref_ver_comps.group(2))
        #print("compare: %s vs %s = %d" % (self, ref, r))
        return r

def __str__(self):
    return str(self.epoch) + ":" + self.version

def parse_version(versionstr):
    epoch = 0
    # check for epoch
    m = re.match('([0-9]*):(.*)', versionstr)
    if m:
        (epochstr, versionstr) = m.groups()
        epoch = int(epochstr)
    return Version(epoch, versionstr)

class Package(object):
    """A class for creating objects to manipulate (e.g. create) opkg
    packages."""

    # fn: Package file path
    # relpath: If this argument is set, the file path is given relative to this
    # path when a string representation of the Package object is created. If
    # this argument is not set, the basename of the file path is given.
    def __init__(self, fn=None, relpath=None):
        self.package = None
        self.version = 'none'
        self.parsed_version = None
        self.architecture = None
        self.maintainer = None
        self.source = None
        self.description = None
        self.depends = None
        self.provides = None
        self.replaces = None
        self.conflicts = None
        self.recommends = None
        self.suggests = None
        self.section = None
        self.filename_header = None
        self.file_list = []
        # md5 and size is lazy attribute, computed on demand

```

```

#self.md5 = None
#self.size = None
self.installed_size = None
self.filename = None
self.file_ext_opk = "ipk"
self.homepage = None
self.oe = None
self.priority = None
self.tags = None
self.fn = fn
self.license = None

if fn:
    # see if it is deb format
    f = open(fn, "rb")

    if relpath:
        self.filename = os.path.relpath(fn, relpath)
    else:
        self.filename = os.path.basename(fn)

    ## sys.stderr.write(" extracting control.tar.gz from %s\n"% (fn,))

    ar = arfile.ArFile(f, fn)
    tarStream = ar.open("control.tar.gz")
    tarf = tarfile.open("control.tar.gz", "r", tarStream)
    try:
        control = tarf.extractfile("control")
    except KeyError:
        control = tarf.extractfile("./control")
    try:
        self.read_control(control)
    except TypeError as e:
        sys.stderr.write("Cannot read control file '%s' - %s\n" % (fn, e))
    control.close()
self.scratch_dir = None
self.file_dir = None
self.meta_dir = None

def __getattr__(self, name):
    if name == "md5":
        self._computeFileMD5()
        return self.md5
    elif name == 'size':
        return self._get_file_size()
    else:
        raise AttributeError(name)

def _computeFileMD5(self):
    # compute the MD5.
    if not self.fn:
        self.md5 = 'Unknown'
    else:
        f = open(self.fn, "rb")
        sum = hashlib.md5()

```

```

        while True:
            data = f.read(1024)
            if not data: break
            sum.update(data)
        f.close()
        self.md5 = sum.hexdigest()

    def _get_file_size(self):
        if not self.fn:
            self.size = 0;
        else:
            stat = os.stat(self.fn)
            self.size = stat[ST_SIZE]
        return int(self.size)

    def read_control(self, control):
        import os

        line = control.readline()
        while 1:
            if not line: break
            # Decode if stream has byte strings
            if not isinstance(line, str):
                line = line.decode()
            line = line.rstrip()
            lineparts = re.match(r'([\w-]*?):s*(.*)', line)
            if lineparts:
                name = lineparts.group(1).lower()
                value = lineparts.group(2)
                while 1:
                    line = control.readline()
                    if not line: break
                    if line[0] != ' ': break
                    value = value + '\n' + line
                if name == 'size':
                    self.size = int(value)
                elif name == 'md5sum':
                    self.md5 = value
                elif name in self.__dict__:
                    self.__dict__[name] = value
                else:
                    print("Lost field %s, %s" % (name,value))
                    pass

            if line and line[0] == '\n':
                return # consumes one blank line at end of package descriptoin
            else:
                line = control.readline()
                pass
        return

    def _setup_scratch_area(self):
        self.scratch_dir = "%s/%sopkg" % (tempfile.gettempdir(),
                                           tempfile.gettemprefix())

```

```
self.file_dir = "%s/files" % (self.scratch_dir)
self.meta_dir = "%s/meta" % (self.scratch_dir)

os.mkdir(self.scratch_dir)
os.mkdir(self.file_dir)
os.mkdir(self.meta_dir)

def set_package(self, package):
    self.package = package

def get_package(self):
    return self.package

def set_version(self, version):
    self.version = version
    self.parsed_version = parse_version(version)

def get_version(self):
    return self.version

def set_architecture(self, architecture):
    self.architecture = architecture

def get_architecture(self):
    return self.architecture

def set_maintainer(self, maintainer):
    self.maintainer = maintainer

def get_maintainer(self):
    return self.maintainer

def set_source(self, source):
    self.source = source

def get_source(self):
    return self.source

def set_description(self, description):
    self.description = description

def get_description(self):
    return self.description

def set_depends(self, depends):
    self.depends = depends

def get_depends(self, depends):
    return self.depends

def set_provides(self, provides):
    self.provides = provides

def get_provides(self, provides):
    return self.provides
```



```
def set_replaces(self, replaces):
    self.replaces = replaces

def get_replaces(self, replaces):
    return self.replaces

def set_conflicts(self, conflicts):
    self.conflicts = conflicts

def get_conflicts(self, conflicts):
    return self.conflicts

def set_suggests(self, suggests):
    self.suggests = suggests

def get_suggests(self, suggests):
    return self.suggests

def set_section(self, section):
    self.section = section

def get_section(self, section):
    return self.section

def set_license(self, license):
    self.license = license

def get_license(self, license):
    return self.license

def get_file_list_dir(self, directory):
    def check_output(*popenargs, **kwargs):
        """Run command with arguments and return its output as a byte string.

        Backported from Python 2.7 as it's implemented as pure python on stdlib.

        >>> check_output(['/usr/bin/python', '--version'])
        Python 2.6.2
        """
        process = subprocess.Popen(stdout=subprocess.PIPE, *popenargs, **kwargs)
        output, unused_err = process.communicate()
        retcode = process.poll()
        if retcode:
            cmd = kwargs.get("args")
            if cmd is None:
                cmd = popenargs[0]
            error = subprocess.CalledProcessError(retcode, cmd)
            error.output = output
            raise error
        return output

    if not self.fn:
        try:
```

```

        cmd = "find %s -name %s | head -n 1" % (directory, self.filename)
        rc = check_output(cmd, shell=True)
        if rc != "":
            newfn = str(rc).split()[0]
#            sys.stderr.write("Package '%s' with empty fn and filename is '%s' was found in '%s', updating fn\n" %
(self.package, self.filename, newfn))
            self.fn = newfn
        except OSError as e:
            sys.stderr.write("Cannot find current fn for package '%s' filename '%s' in dir '%s'\n(%s)\n" % (self.pack-
age, self.filename, directory, e))
        except IOError as e:
            sys.stderr.write("Cannot find current fn for package '%s' filename '%s' in dir '%s'\n(%s)\n" % (self.pack-
age, self.filename, directory, e))
        return self.get_file_list()

def get_file_list(self):
    if not self.fn:
        sys.stderr.write("Package '%s' has empty fn, returning empty filelist\n" % (self.package))
        return []
    f = open(self.fn, "rb")
    ar = arfile.ArFile(f, self.fn)
    tarStream = ar.open("data.tar.gz")
    tarf = tarfile.open("data.tar.gz", "r", tarStream)
    self.file_list = tarf.getnames()
    self.file_list = ["/" + a for a in self.file_list if a.startswith("/")]

    f.close()
    return self.file_list

def set_package_extension(self, ext="ipk"):
    self.file_ext_opk = ext

def get_package_extension(self):
    return self.file_ext_opk

def write_package(self, dirname):
    self._setup_scratch_area()
    file = open("%s/control" % self.meta_dir, 'w')
    file.write(str(self))
    file.close()

    cmd = "cd %s ; tar cvz --format=gnu -f %s/control.tar.gz control" % (self.meta_dir,
                                                                    self.scratch_dir)

    cmd_out, cmd_in, cmd_err = os.popen3(cmd)

    while cmd_err.readline() != "":
        pass

    cmd_out.close()
    cmd_in.close()
    cmd_err.close()

    bits = "control.tar.gz"

```

```

if self.file_list:
    cmd = "cd %s ; tar cvz --format=gnu -f %s/data.tar.gz" % (self.file_dir,
                                                             self.scratch_dir)

    cmd_out, cmd_in, cmd_err = os.popen3(cmd)

    while cmd_err.readline() != "":
        pass

    cmd_out.close()
    cmd_in.close()
    cmd_err.close()

    bits = bits + " data.tar.gz"

file = "%s_%s_%s.%s" % (self.package, self.version, self.architecture, self.get_package_extension())
cmd = "cd %s ; tar cvz --format=gnu -f %s/%s %s" % (self.scratch_dir,
                                                    dirname,
                                                    file,
                                                    bits)

cmd_out, cmd_in, cmd_err = os.popen3(cmd)

while cmd_err.readline() != "":
    pass

cmd_out.close()
cmd_in.close()
cmd_err.close()

def compare_version(self, ref):
    """Compare package versions of self and ref"""
    if not self.version:
        print('No version for package %s' % self.package)
    if not ref.version:
        print('No version for package %s' % ref.package)
    if not self.parsed_version:
        self.parsed_version = parse_version(self.version)
    if not ref.parsed_version:
        ref.parsed_version = parse_version(ref.version)
    return self.parsed_version.compare(ref.parsed_version)

def __str__(self):
    out = ""

    # XXX - Some checks need to be made, and some exceptions
    #       need to be thrown. -- a7r

    if self.package: out = out + "Package: %s\n" % (self.package)
    if self.version: out = out + "Version: %s\n" % (self.version)
    if self.depends: out = out + "Depends: %s\n" % (self.depends)
    if self.provides: out = out + "Provides: %s\n" % (self.provides)
    if self.replaces: out = out + "Replaces: %s\n" % (self.replaces)

```

```

        if self.conflicts: out = out + "Conflicts: %s\n" % (self.conflicts)
        if self.suggests: out = out + "Suggests: %s\n" % (self.suggests)
        if self.recommends: out = out + "Recommends: %s\n" % (self.recommends)
        if self.section: out = out + "Section: %s\n" % (self.section)
        if self.architecture: out = out + "Architecture: %s\n" % (self.architecture)
        if self.maintainer: out = out + "Maintainer: %s\n" % (self.maintainer)
        if self.md5: out = out + "MD5Sum: %s\n" % (self.md5)
        if self.size: out = out + "Size: %d\n" % int(self.size)
        if self.installed_size: out = out + "InstalledSize: %d\n" % int(self.installed_size)
        if self.filename: out = out + "Filename: %s\n" % (self.filename)
        if self.source: out = out + "Source: %s\n" % (self.source)
        if self.description:
            printable_description = textwrap.dedent(self.description).strip()
            out = out + "Description: %s\n" % textwrap.fill(printable_description, width=74, initial_indent=' ', subsequent_indent=' ')
        if self.oe: out = out + "OE: %s\n" % (self.oe)
        if self.homepage: out = out + "HomePage: %s\n" % (self.homepage)
        if self.license: out = out + "License: %s\n" % (self.license)
        if self.priority: out = out + "Priority: %s\n" % (self.priority)
        if self.tags: out = out + "Tags: %s\n" % (self.tags)
        out = out + "\n"

    return out

def __del__(self):
    # XXX - Why is the `os` module being yanked out before Package objects
    #       are being destroyed? -- a7r
    pass

class Packages(object):
    """A currently unimplemented wrapper around the opkg utility."""
    def __init__(self):
        self.packages = {}
        return

    def add_package(self, pkg):
        package = pkg.package
        arch = pkg.architecture
        name = ("%s:%s" % (package, arch))
        if (name not in self.packages):
            self.packages[name] = pkg

        if pkg.compare_version(self.packages[name]) >= 0:
            self.packages[name] = pkg
            return 0
        else:
            return 1

    def read_packages_file(self, fn):
        f = open(fn, "r")
        while True:
            pkg = Package()
            try:
                pkg.read_control(f)
            except TypeError as e:

```

```

        sys.stderr.write("Cannot read control file '%s' - %s\n" % (fn, e))
        continue
    if pkg.get_package():
        self.add_package(pkg)
    else:
        break
f.close()
return

def write_packages_file(self, fn):
    f = open(fn, "w")
    names = list(self.packages.keys())
    names.sort()
    for name in names:
        f.write(self.packages[name].__str__())
    return

def keys(self):
    return list(self.packages.keys())

def __getitem__(self, key):
    return self.packages[key]

if __name__ == "__main__":

    assert Version(0, "1.2.2-r1").compare(Version(0, "1.2.3-r0")) == -1
    assert Version(0, "1.2.2-r0").compare(Version(0, "1.2.2+cv20070308-r0")) == -1
    assert Version(0, "1.2.2+cv20070308").compare(Version(0, "1.2.2-r0")) == 1
    assert Version(0, "1.2.2-r0").compare(Version(0, "1.2.2-r0")) == 0
    assert Version(0, "1.2.2-r5").compare(Version(0, "1.2.2-r0")) == 1

    package = Package()

    package.set_package("FooBar")
    package.set_version("0.1-fam1")
    package.set_architecture("arm")
    package.set_maintainer("Testing <testing@testing.testing>")
    package.set_depends("libc")
    package.set_description("A test of the APIs. And very long descriptions so often used in oe-core\nfoo\n\nnbar")

    print("<")
    sys.stdout.write(str(package))
    print(">")
    f = open("/tmp/control", "w")
    f.write(str(package))
    f.close()

    f = open("/tmp/control", "r")
    package2 = Package()
    package2.read_control(f)
    print("<")
    sys.stdout.write(str(package2))
    print(">")

    package.write_package("/tmp")

```

1.106 oprofile

1.106.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

```
/**
 * @file opagent.h
 * Interface to report symbol names and dynamically generated code to Oprofile
 *
 * @remark Copyright 2007 OProfile authors
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 *
 * @author Jens Wilke
 * @Modifications Daniel Hansel
 *
 * Copyright IBM Corporation 2007
 */

#ifndef _LIB_OPAGENT_H
#define _LIB_OPAGENT_H

#include <sys/types.h>
#include <stdint.h>

#ifdef __cplusplus
```

```

extern "C" {
#endif

struct debug_line_info {
    unsigned long vma;
    unsigned int lineno;
    /* The filename format is unspecified, absolute path, relative etc. */
    char const * filename;
};

typedef void * op_agent_t;

/**
 * This function must be called by agents before any other function.
 * Creates and opens a JIT dump file in /tmp/.oprofile/jitdump
 * using the naming convention <process_id>.dump.
 *
 * Returns a valid op_agent_t handle or NULL. If NULL is returned, errno
 * is set to indicate the nature of the error.
 */
op_agent_t op_open_agent(void);

/**
 * Frees all resources and closes open file handles.
 *
 * hdl:      Handle returned from an earlier call to op_open_agent()
 *
 * Returns 0 on success; -1 otherwise. If -1 is returned, errno is
 * set to indicate the nature of the error.
 */
int op_close_agent(op_agent_t hdl);

/**
 * Signal the dynamic generation of native code from a virtual machine.
 * Writes a JIT dump record to the open JIT dump file using
 * the passed information.
 *
 * hdl:      Handle returned from an earlier call to op_open_agent()
 * symbol_name: The name of the symbol being dynamically compiled.
 *             This name can (and should) contain all necessary
 *             information to disambiguate it from symbols of the
 *             same name; e.g., class, method signature.
 * vma:      The virtual memory address of the executable code.
 * code:     Pointer to the location of the compiled code.
 * Theoretically, this may be a different location from
 * that given by the vma argument. For some JIT compilers,
 * obtaining the code may be impractical. For this (or any other)
 * reason, the agent can choose to pass NULL for this parameter.
 * If NULL is passed, no code will be copied into the JIT dump
 * file.
 * code_size: Size of the compiled code.
 *
 * Returns 0 on success; -1 otherwise. If -1 is returned, errno is
 * set to indicate the nature of the error.

```

```

    */
    int op_write_native_code(op_agent_t hdl, char const * symbol_name,
        uint64_t vma, void const * code,
        const unsigned int code_size);

/**
 * Add debug line information to a piece of code. An op_write_native_code()
 * with the same code pointer should have occurred before this call. It's not
 * necessary to provide one lineno information entry per machine instruction;
 * the array can contain hole.
 *
 * hdl:      Handle returned from an earlier call to op_open_agent()
 * code:     Pointer to the location of the code with debug info
 * nr_entry: Number of entries in compile_map
 * compile_map: Array of struct debug_line_info. See the JVMTI agent
 *               library implementation for an example of what information
 *               should be retrieved from a VM to fill out this data structure.
 *
 * Returns 0 on success; -1 otherwise. If -1 is returned, errno is
 * set to indicate the nature of the error.
 */
    int op_write_debug_line_info(op_agent_t hdl, void const * code,
        size_t nr_entry,
        struct debug_line_info const * compile_map);

/**
 * Signal the invalidation of native code from a virtual machine.
 *
 * hdl:      Handle returned from an earlier call to op_open_agent()
 * vma:     The virtual memory address of the compiled code being unloaded.
 *           An op_write_native_code() with the same vma should have
 *           occurred before this call.
 *
 * Returns 0 on success; -1 otherwise. If -1 is returned, errno is
 * set to indicate the nature of the error.
 */
    int op_unload_native_code(op_agent_t hdl, uint64_t vma);

/**
 * Returns the major version number of the libopagent library that will be used.
 */
    int op_major_version(void);

/**
 * Returns the minor version number of the libopagent library that will be used.
 */
    int op_minor_version(void);

/* idea how to post additional information for a piece of code.
   we use the code address as reference
    int op_write_loader_name(const void* code_addr, char const * loader_name);
    */

#ifdef __cplusplus
}

```

#endif

#endif

1.107 pam-radius-auth

1.107.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

/*

*

*RADIUS

*Remote Authentication Dial In User Service

*

*

*Livingston Enterprises, Inc.

*6920 Koll Center Parkway

*Pleasanton, CA 94566

*

*Copyright 1992 Livingston Enterprises, Inc.

*

*Permission to use, copy, modify, and distribute this software for any

*purpose and without fee is hereby granted, provided that this

*copyright and permission notice appear on all copies and supporting

*documentation, the name of Livingston Enterprises, Inc. not be used

*in advertising or publicity pertaining to distribution of the

*program without specific prior permission, and notice be given

*in supporting documentation that copying and distribution is by

*permission of Livingston Enterprises, Inc.

*

*Livingston Enterprises, Inc. makes no representations about

*the suitability of this software for any purpose. It is

*provided "as is" without express or implied warranty.

*

*/

1.108 pciutils

1.108.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.109 perl

1.109.1 Available under license

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications

derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.

3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:

- a) place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
- b) use the modified Package only within your corporation or organization.
- c) rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.
- d) make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

- a) distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.
- b) accompany the distribution with the machine-readable source of the Package with your modifications.
- c) give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
- d) make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this

Package via the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

GNU GENERAL PUBLIC LICENSE
Version 1, February 1989

Copyright (C) 1989 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The license agreements of most software companies try to keep users at the mercy of those companies. By contrast, our General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. The General Public License applies to the Free Software Foundation's software and to any other program whose authors commit to using it. You can use it for your programs, too.

When we speak of free software, we are referring to freedom, not price. Specifically, the General Public License is designed to make sure that you have the freedom to give away or sell copies of free software, that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free

programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of a such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must tell them their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any work containing the Program or a portion of it, either verbatim or with modifications. Each licensee is addressed as "you".

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this General Public License and to the absence of any warranty; and give any other recipients of the Program a copy of this General Public License along with the Program. You may charge a fee for the physical act of transferring a copy.

2. You may modify your copy or copies of the Program or any portion of it, and copy and distribute such modifications under the terms of Paragraph 1 above, provided that you also do the following:

- a) cause the modified files to carry prominent notices stating that you changed the files and the date of any change; and
- b) cause the whole of any work that you distribute or publish, that in whole or in part contains the Program or any part thereof, either

with or without modifications, to be licensed at no charge to all third parties under the terms of this General Public License (except that you may choose to grant warranty protection to some or all third parties, at your option).

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the simplest and most usual way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this General Public License.

d) You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Mere aggregation of another independent work with the Program (or its derivative) on a volume of a storage or distribution medium does not bring the other work under the scope of these terms.

3. You may copy and distribute the Program (or a portion or derivative of it, under Paragraph 2) in object code or executable form under the terms of Paragraphs 1 and 2 above provided that you also do one of the following:

a) accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Paragraphs 1 and 2 above; or,

b) accompany it with a written offer, valid for at least three years, to give any third party free (except for a nominal charge for the cost of distribution) a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Paragraphs 1 and 2 above; or,

c) accompany it with the information you received as to where the corresponding source code may be obtained. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form alone.)

Source code for a work means the preferred form of the work for making modifications to it. For an executable file, complete source code means all the source code for all modules it contains; but, as a special exception, it need not include source code for modules which are standard libraries that accompany the operating system on which the executable file runs, or for standard header files or definitions files that accompany that operating system.

4. You may not copy, modify, sublicense, distribute or transfer the Program except as expressly provided under this General Public License. Any attempt otherwise to copy, modify, sublicense, distribute or transfer the Program is void, and will automatically terminate your rights to use the Program under this License. However, parties who have received

copies, or rights to use copies, from you under this General Public License will not have their licenses terminated so long as such parties remain in full compliance.

5. By copying, distributing or modifying the Program (or any work based on the Program) you indicate your acceptance of this license to do so, and all its terms and conditions.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.

7. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of the license which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the license, you may choose any version ever published by the Free Software Foundation.

8. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

9. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

10. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE

POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to humanity, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 1, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19xx name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (a program to direct compilers to make passes at assemblers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

That's all there is to it!

1.110 popt

1.110.1 Available under license

Copyright (c) 1998 Red Hat Software

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE X CONSORTIUM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the X Consortium shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from the X Consortium.

1.111 portmap

1.111.1 Available under license

```

/*
 * Check if an address belongs to the local system. Adapted from:
 *
 * pmap_svc.c 1.32 91/03/11 Copyright 1984,1990 Sun Microsystems, Inc.
 * get_myaddress.c 2.1 88/07/29 4.0 RPCSRC.
 */

/*
 * Sun RPC is a product of Sun Microsystems, Inc. and is provided for
 * unrestricted use provided that this legend is included on all tape
 * media and as a part of the software program in whole or part. Users
 * may copy or modify Sun RPC without charge, but are not authorized
 * to license or distribute it to anyone else except as part of a product or
 * program developed by the user or with the express written consent of
 * Sun Microsystems, Inc.
 *
 * SUN RPC IS PROVIDED AS IS WITH NO WARRANTIES OF ANY KIND INCLUDING THE
 * WARRANTIES OF DESIGN, MERCHANTIBILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE, OR ARISING FROM A COURSE OF DEALING, USAGE OR TRADE PRACTICE.
 *
 * Sun RPC is provided with no support and without any obligation on the
 * part of Sun Microsystems, Inc. to assist in its use, correction,
 * modification or enhancement.
 *
 * SUN MICROSYSTEMS, INC. SHALL HAVE NO LIABILITY WITH RESPECT TO THE
 * INFRINGEMENT OF COPYRIGHTS, TRADE SECRETS OR ANY PATENTS BY SUN RPC
 * OR ANY PART THEREOF.
 *
 * In no event will Sun Microsystems, Inc. be liable for any lost revenue
 * or profits or other special, indirect and consequential damages, even if
 * Sun has been advised of the possibility of such damages.
 *
 * Sun Microsystems, Inc.
 * 2550 Garcia Avenue
 * Mountain View, California 94043
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.h>
#include <netdb.h>
#include <netinet/in.h>
#include <net/if.h>
#include <sys/ioctl.h>
#include <syslog.h>
#include <string.h>
#include <unistd.h>

```

```
#ifndef TRUE
#define TRUE 1
#define FALSE 0
#endif

/*
 * With virtual hosting, each hardware network interface can have multiple
 * network addresses. On such machines the number of machine addresses can
 * be surprisingly large.
 */
static int num_local;
static int num_addrs;
static struct in_addr *addrs;

/* grow_addrs - extend list of local interface addresses */

static int grow_addrs(void)
{
    struct in_addr *new_addrs;
    int new_num;

    /*
     * Keep the previous result if we run out of memory. The system would
     * really get hosed if we simply give up.
     */
    new_num = (addrs == 0) ? 1 : num_addrs + num_addrs;
    new_addrs = (struct in_addr *) malloc(sizeof(*addrs) * new_num);
    if (new_addrs == 0) {
        perror("portmap: out of memory");
        return (0);
    } else {
        if (addrs != 0) {
            memcpy((char *) new_addrs, (char *) addrs,
                sizeof(*addrs) * num_addrs);
            free((char *) addrs);
        }
        num_addrs = new_num;
        addrs = new_addrs;
        return (1);
    }
}

/* find_local - find all IP addresses for this host */

static int
find_local(void)
{
    struct ifconf ifc;
    struct ifreq ifreq;
    struct ifreq *ifr;
    struct ifreq *the_end;
    int sock;
    char buf[BUFSIZ];
```

```

/*
 * Get list of network interfaces. We use a huge buffer to allow for the
 * presence of non-IP interfaces.
 */

if ((sock = socket(PF_INET, SOCK_DGRAM, 0)) < 0) {
perror("socket");
return (0);
}
ifc.ifc_len = sizeof(buf);
ifc.ifc_buf = buf;
if (ioctl(sock, SIOCGIFCONF, (char *) &ifc) < 0) {
perror("SIOCGIFCONF");
(void) close(sock);
return (0);
}
/* Get IP address of each active IP network interface. */

the_end = (struct ifreq *) (ifc.ifc_buf + ifc.ifc_len);
num_local = 0;
for (ifr = ifc.ifc_req; ifr < the_end; ifr++) {
if (ifr->ifr_addr.sa_family == AF_INET) { /* IP net interface */
ifreq = *ifr;
if (ioctl(sock, SIOCGIFFLAGS, (char *) &ifreq) < 0) {
perror("SIOCGIFFLAGS");
} else if (ifreq.ifr_flags & IFF_UP) { /* active interface */
if (ioctl(sock, SIOCGIFADDR, (char *) &ifreq) < 0) {
perror("SIOCGIFADDR");
} else {
if (num_local >= num_addrs)
if (grow_addrs() == 0)
break;
addrs[num_local++] = ((struct sockaddr_in *)
& ifreq.ifr_addr)->sin_addr;
}
}
}
}
/* Support for variable-length addresses. */
#ifdef HAS_SA_LEN
ifr = (struct ifreq *) ((caddr_t) ifr
+ ifr->ifr_addr.sa_len - sizeof(struct sockaddr));
#endif
}
(void) close(sock);
return (num_local);
}

/* from_local - determine whether request comes from the local system */

int from_local(struct sockaddr_in *addr)
{
int i;

if (addrs == 0 && find_local() == 0)

```



```
syslog(LOG_ERR, "cannot find any active local network interfaces");
```

```
    for (i = 0; i < num_local; i++) {
if (memcmp((char *) &(addr->sin_addr), (char *) &(addrs[i]),
    sizeof(struct in_addr)) == 0)
        return (TRUE);
    }
    return (FALSE);
}
```

```
#ifdef TEST
```

```
main()
{
    char *inet_ntoa();
    int i;

    find_local();
    for (i = 0; i < num_local; i++)
printf("%s\n", inet_ntoa(addrs[i]));
}
```

```
#endif
```

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```
/*_
```

```
* Copyright (c) 1990 The Regents of the University of California.
```

```
* All rights reserved.
```

```
*
```

```
* Redistribution and use in source and binary forms, with or without
```

```

* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
*   must display the following acknowledgement:
* This product includes software developed by the University of
* California, Berkeley and its contributors.
* 4. Neither the name of the University nor the names of its contributors
*   may be used to endorse or promote products derived from this software
*   without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*/

```

```

#ifndef lint
char copyright[] =
"@(#) Copyright (c) 1990 The Regents of the University of California.\n\
All rights reserved.\n";
#endif /* not lint */

/*
@(#)portmap.c2.3 88/08/11 4.0 RPCSRC
static char sccsid[] = "@(#)portmap.c 1.32 87/08/06 Copyr 1984 Sun Micro";
*/

/*
* portmap.c, Implements the program, version to port number mapping for
* rpc.
*/

/*
* Sun RPC is a product of Sun Microsystems, Inc. and is provided for
* unrestricted use provided that this legend is included on all tape
* media and as a part of the software program in whole or part. Users
* may copy or modify Sun RPC without charge, but are not authorized
* to license or distribute it to anyone else except as part of a product or
* program developed by the user.
*
* SUN RPC IS PROVIDED AS IS WITH NO WARRANTIES OF ANY KIND INCLUDING THE
* WARRANTIES OF DESIGN, MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE, OR ARISING FROM A COURSE OF DEALING, USAGE OR TRADE PRACTICE.

```

```
*
* Sun RPC is provided with no support and without any obligation on the
* part of Sun Microsystems, Inc. to assist in its use, correction,
* modification or enhancement.
*
* SUN MICROSYSTEMS, INC. SHALL HAVE NO LIABILITY WITH RESPECT TO THE
* INFRINGEMENT OF COPYRIGHTS, TRADE SECRETS OR ANY PATENTS BY SUN RPC
* OR ANY PART THEREOF.
*
* In no event will Sun Microsystems, Inc. be liable for any lost revenue
* or profits or other special, indirect and consequential damages, even if
* Sun has been advised of the possibility of such damages.
*
* Sun Microsystems, Inc.
* 2550 Garcia Avenue
* Mountain View, California 94043
*/

#include <rpc/rpc.h>
#include <rpc/pmap_prot.h>
#include <stdio.h>
#include <unistd.h>
#include <syslog.h>
#include <netdb.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <sys/wait.h>
#include <sys/signal.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <sys/types.h>
#ifdef SYSV40
#include <netinet/in.h>
#endif
#include <arpa/inet.h>

#include <stdlib.h>
#include <pwd.h>

#ifndef LOG_PERROR
#define LOG_PERROR 0
#endif

#ifndef LOG_DAEMON
#define LOG_DAEMON 0
#endif

/* Older SYSV. */
#if !defined(SIGCHLD) && defined(SIGCLD)
#define SIGCHLD SIGCLD
#endif
```

```

#ifndef svc_getcaller/* SYSV4 */
# define svc_getcaller svc_getrpcaller
#endif

static void reg_service(struct svc_req *rqstp, SVCXPRT *xprt);
#ifndef IGNORE_SIGCHLD/* Lionel Cons <cons@dxcern.cern.ch> */
static void reap(int);
#endif
static void callit(struct svc_req *rqstp, SVCXPRT *xprt);
struct pmaplist *pmaplist;
int debugging = 0;
int store_fd = -1;
static void dump_table(void);
static void load_table(void);

#include "pmap_check.h"

/*
 * How desperate can one be. It is possible to prevent an attacker from
 * manipulating your portmapper tables from outside with requests that
 * contain spoofed source address information. The countermeasure is to
 * force all rpc servers to register and unregister with the portmapper via
 * the loopback network interface, instead of via the primary network
 * interface that every host can talk to. For this countermeasure to work it
 * is necessary to #define LOOPBACK_SETUNSET, to disable source routing in
 * the kernel, and to modify libc so that get_myaddress() chooses the
 * loopback interface address.
 */

#ifndef LOOPBACK_SETUNSET
static SVCXPRT *ludpxprt, *ltxprt;
static int on = 1;
#ifndef INADDR_LOOPBACK
#define INADDR_LOOPBACK ntohl(inet_addr("127.0.0.1"))
#endif
#endif

#ifndef DAEMON_UID
int daemon_uid = DAEMON_UID;
int daemon_gid = DAEMON_GID;
#else
int daemon_uid = 1;
int daemon_gid = 1;
#endif

/*
 * We record with each registration a flag telling whether it was
 * registered with a privilege port or not.
 * If it was, it can only be unregistered with a privileged port.
 * So that we can still use standard pmap xdr routines, we store
 * this flag in a structure wrapped around a pmaplist.
 */
struct flagged_pml {
struct pmaplist pml;
int priv;

```

```
};

int
main(int argc, char **argv)
{
    SVCXPRT *xpvt;
    int sock, c;
    struct sockaddr_in addr;
    int len = sizeof(struct sockaddr_in);
    struct pmaplist *pml;
    struct flagged_pml *fpml;
    char *chroot_path = NULL;
    struct in_addr bindaddr;
    int have_bindaddr = 0;
    int foreground = 0;
    int have_uid = 0;

    while ((c = getopt(argc, argv, "Vdflt:vi:u:g:")) != EOF) {
        switch (c) {

            case 'V':
                printf("portmap version 6.0 - 2007-May-11\n");
                exit(1);

            case 'u':
                daemon_uid = atoi(optarg);
                if (daemon_uid <= 0) {
                    fprintf(stderr,
                        "portmap: illegal uid: %s\n", optarg);
                    exit(1);
                }
                have_uid = 1;
                break;

            case 'g':
                daemon_gid = atoi(optarg);
                if (daemon_gid <= 0) {
                    fprintf(stderr,
                        "portmap: illegal gid: %s\n", optarg);
                    exit(1);
                }
                have_uid = 1;
                break;

            case 'd':
                debugging = 1;
            case 'f':
                foreground = 1;
                break;

            case 't':
                chroot_path = optarg;
                break;

            case 'v':
                verboselog = 1;
```

```

break;

case 'l':
optarg = "127.0.0.1";
/* FALL THROUGH */
case 'i':
have_bindaddr = inet_aton(optarg, &bindaddr);
break;
default:
fprintf(stderr,
"usage: %s [-dflv] [-t dir] [-i address] "
"[-u uid] [-g gid]\n",
argv[0]);
fprintf(stderr, "-d: debugging mode\n");
fprintf(stderr,
"-f: don't daemonize, log to standard error\n");
fprintf(stderr, "-t dir: chroot into dir\n");
fprintf(stderr, "-v: verbose logging\n");
fprintf(stderr, "-i address: bind to address\n");
fprintf(stderr, "-l: same as -i 127.0.0.1\n");
fprintf(stderr, "-u uid : setuid to this uid\n");
fprintf(stderr, "-g uid : setgid to this gid\n");
exit(1);
}
}

if (!foreground && daemon(0, 0)) {
(void) fprintf(stderr, "portmap: fork: %s", strerror(errno));
exit(1);
}

#ifdef LOG_DAEMON
openlog("portmap", LOG_PID|LOG_NDELAY | ( foreground ? LOG_PERROR : 0),
FACILITY);
#else
openlog("portmap", LOG_PID|LOG_NDELAY | ( foreground ? LOG_PERROR : 0));
#endif

#ifdef RPCUSER
if (!have_uid) {
struct passwd *pwent;
pwent = getpwnam(RPCUSER);
if (pwent) {
daemon_uid = pwent->pw_uid;
daemon_gid = pwent->pw_gid;
} else
syslog(LOG_WARNING, "user "" RPCUSER
"" not found, reverting to default uid");
}
#endif

if ((sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) < 0) {
syslog(LOG_ERR, "cannot create udp socket: %m");
exit(1);
}

```

```

#ifdef LOOPBACK_SETUNSET
setsockopt(sock, SOL_SOCKET, SO_REUSEADDR, &on, sizeof on);
#endif

memset((char *) &addr, 0, sizeof(addr));
addr.sin_addr.s_addr = 0;
addr.sin_family = AF_INET;
addr.sin_port = htons(PMAPPORT);
if (have_bindaddr)
memcpy(&addr.sin_addr, &bindaddr, sizeof(bindaddr));

if (bind(sock, (struct sockaddr *)&addr, len) != 0) {
syslog(LOG_ERR, "cannot bind udp: %m");
exit(1);
}

if ((xpirt = svcudp_create(sock)) == (SVCXPRT *)NULL) {
syslog(LOG_ERR, "couldn't do udp_create");
exit(1);
}
/* make an entry for ourself */
fpml = malloc(sizeof(struct flagged_pml));
pml = &fpml->pml;
fpml->priv = 1;
pml->pml_next = NULL;
pml->pml_map.pm_prog = PMAPPROG;
pml->pml_map.pm_vers = PMAPVERS;
pml->pml_map.pm_prot = IPPROTO_UDP;
pml->pml_map.pm_port = PMAPPORT;
pmaplist = pml;

if ((sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
syslog(LOG_ERR, "cannot create tcp socket: %m");
exit(1);
}
#ifdef LOOPBACK_SETUNSET
setsockopt(sock, SOL_SOCKET, SO_REUSEADDR, &on, sizeof on);
#endif
if (bind(sock, (struct sockaddr *)&addr, len) != 0) {
syslog(LOG_ERR, "cannot bind tcp: %m");
exit(1);
}
if ((xpirt = svctcp_create(sock, RPCSMALLMSGSIZE, RPCSMALLMSGSIZE))
== (SVCXPRT *)NULL) {
syslog(LOG_ERR, "couldn't do tcp_create");
exit(1);
}
/* make an entry for ourself */
fpml = malloc(sizeof(struct flagged_pml));
pml = &fpml->pml;
fpml->priv = 1;
pml->pml_map.pm_prog = PMAPPROG;
pml->pml_map.pm_vers = PMAPVERS;
pml->pml_map.pm_prot = IPPROTO_TCP;

```

```

pml->pml_map.pm_port = PMAPPORT;
pml->pml_next = pmaplist;
pmaplist = pml;

#ifdef LOOPBACK_SETUNSET
if ((sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) < 0) {
    syslog(LOG_ERR, "cannot create udp socket: %m");
    exit(1);
}
setsockopt(sock, SOL_SOCKET, SO_REUSEADDR, &on, sizeof on);

addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
if (bind(sock, (struct sockaddr *)&addr, len) != 0) {
    syslog(LOG_ERR, "cannot bind udp: %m");
    exit(1);
}

if ((ludpxprt = svcudp_create(sock)) == (SVCXPRT *)NULL) {
    syslog(LOG_ERR, "couldn't do udp_create");
    exit(1);
}

if ((sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
    syslog(LOG_ERR, "cannot create tcp socket: %m");
    exit(1);
}
setsockopt(sock, SOL_SOCKET, SO_REUSEADDR, &on, sizeof on);
if (bind(sock, (struct sockaddr *)&addr, len) != 0) {
    syslog(LOG_ERR, "cannot bind tcp: %m");
    exit(1);
}
if ((ltcpxpirt = svctcp_create(sock, RPCSMALLMSGSIZE, RPCSMALLMSGSIZE))
    == (SVCXPRT *)NULL) {
    syslog(LOG_ERR, "couldn't do tcp_create");
    exit(1);
}
#endif

(void)svc_register(xprt, PMAPPROG, PMAPVERS, reg_service, FALSE);

store_fd = open("/var/run/portmap_mapping", O_RDWR|O_CREAT, 0600);
load_table();

/* additional initializations */
if (chroot_path) {
    if (chroot(chroot_path) < 0) {
        syslog(LOG_ERR, "couldn't do chroot");
        exit(1);
    }
}

check_startup();
#ifdef IGNORE_SIGCHLD /* Lionel Cons <cons@dxcern.cern.ch> */
(void)signal(SIGCHLD, SIG_IGN);
#else
(void)signal(SIGCHLD, reap);
#endif

```



```
(void)signal(SIGPIPE, SIG_IGN);
svc_run();
syslog(LOG_ERR, "run_svc returned unexpectedly");
abort();
}

/* need to override perror calls in rpc library */
void perror(const char *what)
{

syslog(LOG_ERR, "%s: %m", what);
}

static struct pmaplist *
find_service(u_long prog, u_long vers, u_long prot)
{
struct pmaplist *hit = NULL;
struct pmaplist *pml;

for (pml = pmaplist; pml != NULL; pml = pml->pml_next) {
if ((pml->pml_map.pm_prog != prog) ||
(pml->pml_map.pm_prot != prot))
continue;
hit = pml;
if (pml->pml_map.pm_vers == vers)
break;
}
return (hit);
}

/*
 * 1 OK, 0 not
 */
static void reg_service(struct svc_req *rqstp, SVCXPRT *xpirt)
{
struct pmap reg;
struct pmaplist *pml, *prevpml, *fnd;
struct flagged_pml *fpml;
int ans, port;
caddr_t t;

/*
 * Later wrappers change the logging severity on the fly. Reset to
 * defaults before handling the next request.
 */
allow_severity = LOG_INFO;
deny_severity = LOG_WARNING;

if (debugging)
(void) fprintf(stderr, "server: about do a switch\n");
switch (rqstp->rq_proc) {

case PMAPPROC_NULL:
/*
```

```

    * Null proc call
    */
    /* remote host authorization check */
    check_default(svc_getcaller(xprt), rqstp->rq_proc, (u_long) 0);
    if (!svc_sendreply(xprt, (xdrproc_t) xdr_void, (caddr_t)0)
        && debugging) {
        abort();
    }
    break;

case PMAPPROC_SET:
    /*
     * Set a program,version to port mapping
     */
    if (!svc_getargs(xprt, (xdrproc_t) xdr_pmap, (caddr_t)&reg))
        svcerr_decode(xprt);
    else {
        /* reject non-local requests, protect priv. ports */
        if (!CHECK_SETUNSET(xprt, ludpxprt, ltcpxpirt,
            rqstp->rq_proc, reg.pm_prog, reg.pm_port)) {
            ans = 0;
            goto done;
        }
        /*
         * check to see if already used
         * find_service returns a hit even if
         * the versions don't match, so check for it
         */
        fnd = find_service(reg.pm_prog, reg.pm_vers, reg.pm_prot);
        if (fnd && fnd->pml_map.pm_vers == reg.pm_vers) {
            if (fnd->pml_map.pm_port == reg.pm_port) {
                ans = 1;
                goto done;
            }
        } else {
            ans = 0;
            goto done;
        }
    } else {
        } else {
    /*
     * add to END of list
     */
    fpml = (struct flagged_pml *)
        malloc((u_int)sizeof(struct flagged_pml));
    pml = &fpml->pml;
    fpml->priv =
        (ntohs(svc_getcaller(xprt)->sin_port)
        < IPPORT_RESERVED);
    pml->pml_map = reg;
    pml->pml_next = 0;

    if (pmaplist == 0) {
        pmaplist = pml;
    } else {
        for (fnd= pmaplist; fnd->pml_next != 0;

```

```

        fnd = fnd->pml_next);
fnd->pml_next = pml;
}
ans = 1;
dump_table();
}
done:
if ((!svc_sendreply(xprt, (xdrproc_t)xdr_int,
    (caddr_t)&ans)) &&
    debugging) {
(void) fprintf(stderr, "svc_sendreply\n");
abort();
}
}
break;

case PMAPPROC_UNSET:
/*
 * Remove a program,version to port mapping.
 */
if (!svc_getargs(xprt, (xdrproc_t)xdr_pmap, (caddr_t)&reg))
    svcerr_decode(xprt);
else {
    ans = 0;
    /* reject non-local requests */
    if (!CHECK_SETUNSET(xprt, ludpxprt, ltcpxprt,
        rqstp->rq_proc, reg.pm_prog, (u_long) 0))
        goto done;
    for (prevpml = NULL, pml = pmaplist; pml != NULL; ) {
        if ((pml->pml_map.pm_prog != reg.pm_prog) ||
            (pml->pml_map.pm_vers != reg.pm_vers)) {
            /* both pml & prevpml move forwards */
            prevpml = pml;
            pml = pml->pml_next;
            continue;
        }
        /* found it; pml moves forward, prevpml stays */
        /* privileged port check */
        if (!check_privileged_port(svc_getcaller(xprt),
            rqstp->rq_proc,
            reg.pm_prog,
            pml->pml_map.pm_port)) {
            ans = 0;
            break;
        }
        fpml = (struct flagged_pml*)pml;
        if (fpml->priv &&
            (ntohs(svc_getcaller(xprt)->sin_port)
                >= IPPORT_RESERVED)) {
            ans = 0;
            break;
        }
    }
    ans = 1;

```

```

t = (caddr_t)pml;
pml = pml->pml_next;
if (prevpml == NULL)
    pmaplist = pml;
else
    prevpml->pml_next = pml;
free(t);
dump_table();
}
if (((!svc_sendreply(xprt, (xdrproc_t)xdr_int,
    (caddr_t)&ans)) &&
    debugging) {
    (void) fprintf(stderr, "svc_sendreply\n");
    abort();
}
}
break;

case PMAPPROC_GETPORT:
/*
 * Lookup the mapping for a program,version and return its port
 */
if (!svc_getargs(xprt, (xdrproc_t)xdr_pmap, (caddr_t)&reg))
    svcerr_decode(xprt);
else {
    /* remote host authorization check */
    if (!check_default(svc_getcaller(xprt),
        rqstp->rq_proc,
        reg.pm_prog)) {
        ans = 0;
        goto done;
    }
    fnd = find_service(reg.pm_prog, reg.pm_vers, reg.pm_prot);
    if (fnd)
        port = fnd->pml_map.pm_port;
    else
        port = 0;
    if (((!svc_sendreply(xprt, (xdrproc_t)xdr_int,
        (caddr_t)&port)) &&
        debugging) {
        (void) fprintf(stderr, "svc_sendreply\n");
        abort();
    }
}
break;

case PMAPPROC_DUMP:
/*
 * Return the current set of mapped program,version
 */
if (!svc_getargs(xprt, (xdrproc_t)xdr_void, NULL))
    svcerr_decode(xprt);
else {
    /* remote host authorization check */
    struct pmaplist *p;

```

```

    if (!check_default(svc_getcaller(xprt),
        rqstp->rq_proc, (u_long) 0)) {
        p = 0; /* send empty list */
    } else {
        p = pmaplist;
    }
    if ((!svc_sendreply(xprt, (xdrproc_t)xdr_pmaplist,
        (caddr_t)&p)) && debugging) {
        (void) fprintf(stderr, "svc_sendreply\n");
        abort();
    }
    break;

case PMAPPROC_CALLIT:
    /*
     * Calls a procedure on the local machine. If the requested
     * procedure is not registered this procedure does not return
     * error information!!
     * This procedure is only supported on rpc/udp and calls via
     * rpc/udp. It passes null authentication parameters.
     */
    callit(rqstp, xprt);
    break;

default:
    /* remote host authorization check */
    check_default(svc_getcaller(xprt), rqstp->rq_proc, (u_long) 0);
    svcerr_noproc(xprt);
    break;
}

/*
 * Stuff for the rmtcall service
 */
#define ARGSIZE 9000

struct encap_parms {
    u_int arglen;
    char *args;
};

static bool_t
xdr_encap_parms(XDR *xdrs, struct encap_parms *epp)
{
    return (xdr_bytes(xdrs, &(epp->args), &(epp->arglen), ARGSIZE));
}

struct rmtcallargs {
    u_longrmt_prog;
    u_longrmt_vers;

```

```

u_longrmt_port;
u_longrmt_proc;
struct encap_parms rmt_args;
};

static bool_t
xdr_rmtcall_args(XDR *xdrs, struct rmtcallargs *cap)
{
    /* does not get a port number */
    if (xdr_u_long(xdrs, &(cap->rmt_prog)) &&
        xdr_u_long(xdrs, &(cap->rmt_vers)) &&
        xdr_u_long(xdrs, &(cap->rmt_proc))) {
        return (xdr_encap_parms(xdrs, &(cap->rmt_args)));
    }
    return (FALSE);
}

static bool_t
xdr_rmtcall_result(XDR *xdrs, struct rmtcallargs *cap)
{
    if (xdr_u_long(xdrs, &(cap->rmt_port)))
        return (xdr_encap_parms(xdrs, &(cap->rmt_args)));
    return (FALSE);
}

/*
 * only worries about the struct encap_parms part of struct rmtcallargs.
 * The arglen must already be set!!
 */
static bool_t
xdr_opaque_parms(XDR *xdrs, struct rmtcallargs *cap)
{
    return (xdr_opaque(xdrs, cap->rmt_args.args, cap->rmt_args.arglen));
}

/*
 * This routine finds and sets the length of incoming opaque paraters
 * and then calls xdr_opaque_parms.
 */
static bool_t
xdr_len_opaque_parms(XDR *xdrs, struct rmtcallargs *cap)
{
    u_int beginpos, lowpos, highpos, currpos, pos;

    beginpos = lowpos = pos = xdr_getpos(xdrs);
    highpos = lowpos + ARGSize;
    while ((int)(highpos - lowpos) >= 0) {
        currpos = (lowpos + highpos) / 2;
        if (xdr_setpos(xdrs, currpos)) {
            pos = currpos;
            lowpos = currpos + 1;
        } else {
            highpos = currpos - 1;
        }
    }

```

```

    }
    }
    xdr_setpos(xdrs, beginpos);
    cap->rmt_args.arglen = pos - beginpos;
    return (xdr_opaque_parms(xdrs, cap));
}

/*
 * Call a remote procedure service
 * This procedure is very quiet when things go wrong.
 * The proc is written to support broadcast rpc. In the broadcast case,
 * a machine should shut-up instead of complain, less the requestor be
 * overrun with complaints at the expense of not hearing a valid reply ...
 *
 * This now forks so that the program & process that it calls can call
 * back to the portmapper.
 */
static void callit(struct svc_req *rqstp, SVCXPRT *xpvt)
{
    struct rmtcallargs a;
    struct pmaplist *pml;
    u_short port;
    struct sockaddr_in me;
    int pid, so = -1;
    CLIENT *client;
    struct authunix_parms *au = (struct authunix_parms *)rqstp->rq_clntcred;
    struct timeval timeout;
    char buf[ARGSIZE];

    timeout.tv_sec = 5;
    timeout.tv_usec = 0;
    a.rmt_args.args = buf;
    if (!svc_getargs(xpvt, (xdrproc_t)xdr_rmtcall_args, (caddr_t)&a))
        return;
    /* host and service access control */
    if (!check_callit(svc_getcaller(xpvt),
        rqstp->rq_proc, a.rmt_prog, a.rmt_proc))
        return;
    if ((pml = find_service(a.rmt_prog, a.rmt_vers,
        (u_long)IPPROTO_UDP)) == NULL)
        return;
    /*
     * fork a child to do the work. Parent immediately returns.
     * Child exits upon completion.
     */
    if ((pid = fork()) != 0) {
        if (pid < 0)
            syslog(LOG_ERR, "CALLIT (prog %lu): fork: %m",
                a.rmt_prog);
        return;
    }
    port = pml->pml_map.pm_port;
    get_myaddress(&me);
    me.sin_port = htons(port);

```

```

client = clntudp_create(&me, a.rmt_prog, a.rmt_vers, timeout, &so);
if (client != (CLIENT *)NULL) {
if (rqstp->rq_cred.oa_flavor == AUTH_UNIX) {
client->cl_auth = authunix_create(au->aup_machname,
    au->aup_uid, au->aup_gid, au->aup_len, au->aup_gids);
}
a.rmt_port = (u_long)port;
if (clnt_call(client, a.rmt_proc, (xdrproc_t)xdr_opaque_parms,
    (caddr_t)&a, (xdrproc_t)xdr_len_opaque_parms,
    (caddr_t)&a, timeout) == RPC_SUCCESS) {
svc_sendreply(xprt, (xdrproc_t)xdr_rmtcall_result,
    (caddr_t)&a);
}
AUTH_DESTROY(client->cl_auth);
clnt_destroy(client);
}
(void)close(so);
exit(0);
}

#ifdef IGNORE_SIGCHLD /* Lionel Cons <cons@dxcern.cern.ch> */
static void reap(int ignore)
{
int save_errno = errno;
while (wait3((int *)NULL, WNOHANG, (struct rusage *)NULL) > 0);
errno = save_errno;
}
#endif

/* Dump and restore mapping table so that we can survive kill/restart.
 * To cope with chroot, an fd is opened early and we just write to that.
 * If we are killed while writing the file, we lose, but that isn't
 * very likely...
 */

static void dump_table(void)
{
FILE *f;
struct pmaplist *pml;

if (store_fd < 0)
return;
ftruncate(store_fd, 0);
lseek(store_fd, 0, 0);
f = fdopen(dup(store_fd), "w");
if (!f)
return;

for (pml = pmaplist; pml; pml = pml->pml_next) {
struct flagged_pml *fpml = (struct flagged_pml*)pml;

fprintf(f, "%lu %lu %lu %lu %d\n",
    pml->pml_map.pm_prog,
    pml->pml_map.pm_vers,
    pml->pml_map.pm_prot,

```



```
pml->pml_map.pm_port,
fpml->priv);
}
fclose(f);
}

static void load_table(void)
{
FILE *f;
struct pmaplist **ep;
struct flagged_pml fpml, *fpmlp;

ep = &pmaplist;
while ((*ep)->pml_next)
ep = &(*ep)->pml_next;

if (store_fd < 0)
return;
lseek(store_fd, 0, 0);
f = fdopen(dup(store_fd), "r");
if (f == NULL)
return;

while (fscanf(f, "%lu %lu %lu %lu %d\n",
    &fpml.pml.pml_map.pm_prog,
    &fpml.pml.pml_map.pm_vers,
    &fpml.pml.pml_map.pm_prot,
    &fpml.pml.pml_map.pm_port,
    &fpml.priv) == 5) {
if (fpml.pml.pml_map.pm_port == PMAPPORT)
continue;
fpmlp = malloc(sizeof(struct flagged_pml));
if (!fpmlp)
break;
*fpmlp = fpml;
*ep = &fpmlp->pml;
ep = &fpmlp->pml.pml_next;
*ep = NULL;
}
fclose(f);
}
```

1.112 ppp

1.112.1 Available under license

```

/*
 * ccp.c - PPP Compression Control Protocol.
 *
 * Copyright (c) 1994-2002 Paul Mackerras. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. The name(s) of the authors of this software must not be used to
 *    endorse or promote products derived from this software without
 *    prior written permission.
 *
 * 3. Redistributions of any form whatsoever must retain the following
 *    acknowledgment:
 *    "This product includes software developed by Paul Mackerras
 *    <paulus@samba.org>".
 *
 * THE AUTHORS OF THIS SOFTWARE DISCLAIM ALL WARRANTIES WITH REGARD TO
 * THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS, IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY
 * SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
 * WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN
 * AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING
 * OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
 */

#define RCSID"$Id: ccp.c,v 1.50 2005/06/26 19:34:41 carlsonj Exp $"

#include <stdlib.h>
#include <string.h>

#include "pppd.h"
#include "fsm.h"
#include "ccp.h"
#include <net/ppp-comp.h>

#ifdef MPPE
#include "chap_ms.h"/* mppe_xxxx_key, mppe_keys_set */
#include "lcp.h"/* lcp_close(), lcp_fsm */
#endif

static const char rcsid[] = RCSID;

/*

```

```

* Unfortunately there is a bug in zlib which means that using a
* size of 8 (window size = 256) for Deflate compression will cause
* buffer overruns and kernel crashes in the deflate module.
* Until this is fixed we only accept sizes in the range 9 .. 15.
* Thanks to James Carlson for pointing this out.
*/
#define DEFLATE_MIN_WORKS9

/*
* Command-line options.
*/
static int setbsdcomp __P((char **));
static int setdeflate __P((char **));
static char bsd_value[8];
static char deflate_value[8];

/*
* Option variables.
*/
#ifdef MPPE
bool refuse_mppe_stateful = 1; /* Allow stateful mode? */
#endif

static option_t ccp_option_list[] = {
    { "noccp", o_bool, &ccp_protent.enabled_flag,
      "Disable CCP negotiation" },
    { "-ccp", o_bool, &ccp_protent.enabled_flag,
      "Disable CCP negotiation", OPT_ALIAS },

    { "bsdcomp", o_special, (void *)setbsdcomp,
      "Request BSD-Compress packet compression",
      OPT_PRIO | OPT_A2STRVAL | OPT_STATIC, bsd_value },
    { "nobsdcomp", o_bool, &ccp_wantoptions[0].bsd_compress,
      "don't allow BSD-Compress", OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].bsd_compress },
    { "-bsdcomp", o_bool, &ccp_wantoptions[0].bsd_compress,
      "don't allow BSD-Compress", OPT_ALIAS | OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].bsd_compress },

    { "deflate", o_special, (void *)setdeflate,
      "request Deflate compression",
      OPT_PRIO | OPT_A2STRVAL | OPT_STATIC, deflate_value },
    { "nodeflate", o_bool, &ccp_wantoptions[0].deflate,
      "don't allow Deflate compression", OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].deflate },
    { "-deflate", o_bool, &ccp_wantoptions[0].deflate,
      "don't allow Deflate compression", OPT_ALIAS | OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].deflate },

    { "nodeflatedraft", o_bool, &ccp_wantoptions[0].deflate_draft,
      "don't use draft deflate #", OPT_A2COPY,
      &ccp_allowoptions[0].deflate_draft },

    { "predictor1", o_bool, &ccp_wantoptions[0].predictor_1,

```

```

    "request Predictor-1", OPT_PRIO | 1 },
    { "nopredictor1", o_bool, &ccp_wantoptions[0].predictor_1,
      "don't allow Predictor-1", OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].predictor_1 },
    { "-predictor1", o_bool, &ccp_wantoptions[0].predictor_1,
      "don't allow Predictor-1", OPT_ALIAS | OPT_PRIOSUB | OPT_A2CLR,
      &ccp_allowoptions[0].predictor_1 },

```

```
#ifndef MPPE
```

```

/* MPPE options are symmetrical ... we only set wantoptions here */
{ "require-mppe", o_bool, &ccp_wantoptions[0].mppe,
  "require MPPE encryption",
  OPT_PRIO | MPPE_OPT_40 | MPPE_OPT_128 },
{ "+mppe", o_bool, &ccp_wantoptions[0].mppe,
  "require MPPE encryption",
  OPT_ALIAS | OPT_PRIO | MPPE_OPT_40 | MPPE_OPT_128 },
{ "nomppe", o_bool, &ccp_wantoptions[0].mppe,
  "don't allow MPPE encryption", OPT_PRIO },
{ "-mppe", o_bool, &ccp_wantoptions[0].mppe,
  "don't allow MPPE encryption", OPT_ALIAS | OPT_PRIO },

/* We use ccp_allowoptions[0].mppe as a junk var ... it is reset later */
{ "require-mppe-40", o_bool, &ccp_allowoptions[0].mppe,
  "require MPPE 40-bit encryption", OPT_PRIO | OPT_A2OR | MPPE_OPT_40,
  &ccp_wantoptions[0].mppe },
{ "+mppe-40", o_bool, &ccp_allowoptions[0].mppe,
  "require MPPE 40-bit encryption", OPT_PRIO | OPT_A2OR | MPPE_OPT_40,
  &ccp_wantoptions[0].mppe },
{ "nomppe-40", o_bool, &ccp_allowoptions[0].mppe,
  "don't allow MPPE 40-bit encryption",
  OPT_PRIOSUB | OPT_A2CLRB | MPPE_OPT_40, &ccp_wantoptions[0].mppe },
{ "-mppe-40", o_bool, &ccp_allowoptions[0].mppe,
  "don't allow MPPE 40-bit encryption",
  OPT_ALIAS | OPT_PRIOSUB | OPT_A2CLRB | MPPE_OPT_40,
  &ccp_wantoptions[0].mppe },

{ "require-mppe-128", o_bool, &ccp_allowoptions[0].mppe,
  "require MPPE 128-bit encryption", OPT_PRIO | OPT_A2OR | MPPE_OPT_128,
  &ccp_wantoptions[0].mppe },
{ "+mppe-128", o_bool, &ccp_allowoptions[0].mppe,
  "require MPPE 128-bit encryption",
  OPT_ALIAS | OPT_PRIO | OPT_A2OR | MPPE_OPT_128,
  &ccp_wantoptions[0].mppe },
{ "nomppe-128", o_bool, &ccp_allowoptions[0].mppe,
  "don't allow MPPE 128-bit encryption",
  OPT_PRIOSUB | OPT_A2CLRB | MPPE_OPT_128, &ccp_wantoptions[0].mppe },
{ "-mppe-128", o_bool, &ccp_allowoptions[0].mppe,
  "don't allow MPPE 128-bit encryption",
  OPT_ALIAS | OPT_PRIOSUB | OPT_A2CLRB | MPPE_OPT_128,
  &ccp_wantoptions[0].mppe },

/* strange one; we always request stateless, but will we allow stateful? */
{ "mppe-stateful", o_bool, &refuse_mppe_stateful,
  "allow MPPE stateful mode", OPT_PRIO },
{ "nomppe-stateful", o_bool, &refuse_mppe_stateful,

```

```

    "disallow MPPE stateful mode", OPT_PRIO | 1 },
#endif /* MPPE */

    { NULL }
};

/*
 * Protocol entry points from main code.
 */
static void ccp_init __P((int unit));
static void ccp_open __P((int unit));
static void ccp_close __P((int unit, char *));
static void ccp_lowerup __P((int unit));
static void ccp_lowerdown __P((int));
static void ccp_input __P((int unit, u_char *pkt, int len));
static void ccp_protrej __P((int unit));
static int  ccp_printpkt __P((u_char *pkt, int len,
    void (*printer) __P((void *, char *, ...)),
    void *arg));
static void ccp_datainput __P((int unit, u_char *pkt, int len));

struct protent ccp_protent = {
    PPP_CCP,
    ccp_init,
    ccp_input,
    ccp_protrej,
    ccp_lowerup,
    ccp_lowerdown,
    ccp_open,
    ccp_close,
    ccp_printpkt,
    ccp_datainput,
    1,
    "CCP",
    "Compressed",
    ccp_option_list,
    NULL,
    NULL,
    NULL
};

fsm ccp_fsm[NUM_PPP];
ccp_options ccp_wantoptions[NUM_PPP];/* what to request the peer to use */
ccp_options ccp_gotoptions[NUM_PPP];/* what the peer agreed to do */
ccp_options ccp_allowoptions[NUM_PPP];/* what we'll agree to do */
ccp_options ccp_hisoptions[NUM_PPP];/* what we agreed to do */

/*
 * Callbacks for fsm code.
 */
static void ccp_resetci __P((fsm *));
static int  ccp_cilen __P((fsm *));
static void ccp_addci __P((fsm *, u_char *, int *));
static int  ccp_ackci __P((fsm *, u_char *, int));

```

```

static int ccp_nakci __P((fsm *, u_char *, int, int));
static int ccp_rejci __P((fsm *, u_char *, int));
static int ccp_reqci __P((fsm *, u_char *, int *, int));
static void ccp_up __P((fsm *));
static void ccp_down __P((fsm *));
static int ccp_extcode __P((fsm *, int, int, u_char *, int));
static void ccp_rack_timeout __P((void *));
static char *method_name __P((ccp_options *, ccp_options *));

static fsm_callbacks ccp_callbacks = {
    ccp_resetci,
    ccp_cilen,
    ccp_addci,
    ccp_ackci,
    ccp_nakci,
    ccp_rejci,
    ccp_reqci,
    ccp_up,
    ccp_down,
    NULL,
    NULL,
    NULL,
    NULL,
    ccp_extcode,
    "CCP"
};

/*
 * Do we want / did we get any compression?
 */
#define ANY_COMPRESS(opt)((opt).deflate || (opt).bsd_compress \
|| (opt).predictor_1 || (opt).predictor_2 \
|| (opt).mppe)

/*
 * Local state (mainly for handling reset-reqs and reset-acks).
 */
static int ccp_localstate[NUM_PPP];
#define RACK_PENDING1/* waiting for reset-ack */
#define RREQ_REPEAT2/* send another reset-req if no reset-ack */

#define RACKTIMEOUT1/* second */

static int all_rejected[NUM_PPP];/* we rejected all peer's options */

/*
 * Option parsing.
 */
static int
setbsdcomp(argv)
    char **argv;
{
    int rbits, abits;
    char *str, *endp;

```

```

    str = *argv;
    abits = rbits = strtol(str, &endp, 0);
    if (endp != str && *endp == ',') {
str = endp + 1;
abits = strtol(str, &endp, 0);
    }
    if (*endp != 0 || endp == str) {
option_error("invalid parameter '%s' for bsdcomp option", *argv);
return 0;
    }
    if ((rbits != 0 && (rbits < BSD_MIN_BITS || rbits > BSD_MAX_BITS))
|| (abits != 0 && (abits < BSD_MIN_BITS || abits > BSD_MAX_BITS))) {
option_error("bsdcomp option values must be 0 or %d .. %d",
    BSD_MIN_BITS, BSD_MAX_BITS);
return 0;
    }
    if (rbits > 0) {
ccp_wantoptions[0].bsd_compress = 1;
ccp_wantoptions[0].bsd_bits = rbits;
    } else
ccp_wantoptions[0].bsd_compress = 0;
    if (abits > 0) {
ccp_allowoptions[0].bsd_compress = 1;
ccp_allowoptions[0].bsd_bits = abits;
    } else
ccp_allowoptions[0].bsd_compress = 0;
    sprintf(bsd_value, sizeof(bsd_value),
        rbits == abits? "%d": "%d,%d", rbits, abits);

    return 1;
}

static int
setdeflate(argv)
    char **argv;
{
    int rbits, abits;
    char *str, *endp;

    str = *argv;
    abits = rbits = strtol(str, &endp, 0);
    if (endp != str && *endp == ',') {
str = endp + 1;
abits = strtol(str, &endp, 0);
    }
    if (*endp != 0 || endp == str) {
option_error("invalid parameter '%s' for deflate option", *argv);
return 0;
    }
    if ((rbits != 0 && (rbits < DEFLATE_MIN_SIZE || rbits > DEFLATE_MAX_SIZE))
|| (abits != 0 && (abits < DEFLATE_MIN_SIZE
|| abits > DEFLATE_MAX_SIZE))) {
option_error("deflate option values must be 0 or %d .. %d",
    DEFLATE_MIN_SIZE, DEFLATE_MAX_SIZE);

```

```

return 0;
}
if (rbits == DEFLATE_MIN_SIZE || abits == DEFLATE_MIN_SIZE) {
if (rbits == DEFLATE_MIN_SIZE)
rbits = DEFLATE_MIN_WORKS;
if (abits == DEFLATE_MIN_SIZE)
abits = DEFLATE_MIN_WORKS;
warn("deflate option value of %d changed to %d to avoid zlib bug",
DEFLATE_MIN_SIZE, DEFLATE_MIN_WORKS);
}
if (rbits > 0) {
ccp_wantoptions[0].deflate = 1;
ccp_wantoptions[0].deflate_size = rbits;
} else
ccp_wantoptions[0].deflate = 0;
if (abits > 0) {
ccp_allowoptions[0].deflate = 1;
ccp_allowoptions[0].deflate_size = abits;
} else
ccp_allowoptions[0].deflate = 0;
sprintf(deflate_value, sizeof(deflate_value),
rbits == abits? "%d": "%d,%d", rbits, abits);

return 1;
}

/*
 * ccp_init - initialize CCP.
 */
static void
ccp_init(unit)
    int unit;
{
    fsm *f = &ccp_fsm[unit];

    f->unit = unit;
    f->protocol = PPP_CCP;
    f->callbacks = &ccp_callbacks;
    fsm_init(f);

    memset(&ccp_wantoptions[unit], 0, sizeof(ccp_options));
    memset(&ccp_gotoptions[unit], 0, sizeof(ccp_options));
    memset(&ccp_allowoptions[unit], 0, sizeof(ccp_options));
    memset(&ccp_hisoptions[unit], 0, sizeof(ccp_options));

    ccp_wantoptions[0].deflate = 1;
    ccp_wantoptions[0].deflate_size = DEFLATE_MAX_SIZE;
    ccp_wantoptions[0].deflate_correct = 1;
    ccp_wantoptions[0].deflate_draft = 1;
    ccp_allowoptions[0].deflate = 1;
    ccp_allowoptions[0].deflate_size = DEFLATE_MAX_SIZE;
    ccp_allowoptions[0].deflate_correct = 1;
    ccp_allowoptions[0].deflate_draft = 1;

    ccp_wantoptions[0].bsd_compress = 1;

```



```
    ccp_wantoptions[0].bsd_bits = BSD_MAX_BITS;
    ccp_allowoptions[0].bsd_compress = 1;
    ccp_allowoptions[0].bsd_bits = BSD_MAX_BITS;

    ccp_allowoptions[0].predictor_1 = 1;
}

/*
 * ccp_open - CCP is allowed to come up.
 */
static void
ccp_open(unit)
    int unit;
{
    fsm *f = &ccp_fsm[unit];

    if (f->state != OPENED)
ccp_flags_set(unit, 1, 0);

    /*
     * Find out which compressors the kernel supports before
     * deciding whether to open in silent mode.
     */
    ccp_resetci(f);
    if (!ANY_COMPRESS(ccp_gotoptions[unit]))
f->flags |= OPT_SILENT;

    fsm_open(f);
}

/*
 * ccp_close - Terminate CCP.
 */
static void
ccp_close(unit, reason)
    int unit;
    char *reason;
{
    ccp_flags_set(unit, 0, 0);
    fsm_close(&ccp_fsm[unit], reason);
}

/*
 * ccp_lowerup - we may now transmit CCP packets.
 */
static void
ccp_lowerup(unit)
    int unit;
{
    fsm_lowerup(&ccp_fsm[unit]);
}

/*
 * ccp_lowerdown - we may not transmit CCP packets.
```

```

    */
    static void
    ccp_lowerdown(unit)
        int unit;
    {
        fsm_lowerdown(&ccp_fsm[unit]);
    }

    /*
    * ccp_input - process a received CCP packet.
    */
    static void
    ccp_input(unit, p, len)
        int unit;
        u_char *p;
        int len;
    {
        fsm *f = &ccp_fsm[unit];
        int oldstate;

        /*
        * Check for a terminate-request so we can print a message.
        */
        oldstate = f->state;
        fsm_input(f, p, len);
        if (oldstate == OPENED && p[0] == TERMREQ && f->state != OPENED) {
            notice("Compression disabled by peer.");
#ifdef MPPE
            if (ccp_gotoptions[unit].mppe) {
                error("MPPE disabled, closing LCP");
                lcp_close(unit, "MPPE disabled by peer");
            }
#endif
        }

        /*
        * If we get a terminate-ack and we're not asking for compression,
        * close CCP.
        */
        if (oldstate == REQSENT && p[0] == TERMACK
            && !ANY_COMPRESS(ccp_gotoptions[unit]))
            ccp_close(unit, "No compression negotiated");
    }

    /*
    * Handle a CCP-specific code.
    */
    static int
    ccp_extcode(f, code, id, p, len)
        fsm *f;
        int code, id;
        u_char *p;
        int len;
    {
        switch (code) {

```

```

        case CCP_RESETREQ:
if (f->state != OPENED)
    break;
/* send a reset-ack, which the transmitter will see and
   reset its compression state. */
fsm_sdata(f, CCP_RESETACK, id, NULL, 0);
break;

        case CCP_RESETACK:
if (ccp_localstate[f->unit] & RACK_PENDING && id == f->reqid) {
    ccp_localstate[f->unit] &= ~(RACK_PENDING | RREQ_REPEAT);
    UNTIMEOUT(ccp_rack_timeout, f);
}
break;

        default:
return 0;
    }

    return 1;
}

/*
 * ccp_protrej - peer doesn't talk CCP.
 */
static void
ccp_protrej(unit)
    int unit;
{
    ccp_flags_set(unit, 0, 0);
    fsm_lowerdown(&ccp_fsm[unit]);

#ifdef MPPE
    if (ccp_gotoptions[unit].mppe) {
error("MPPE required but peer negotiation failed");
lcp_close(unit, "MPPE required but peer negotiation failed");
    }
#endif

}

/*
 * ccp_resetci - initialize at start of negotiation.
 */
static void
ccp_resetci(f)
    fsm *f;
{
    ccp_options *go = &ccp_gotoptions[f->unit];
    u_char opt_buf[CCP_MAX_OPTION_LENGTH];

    *go = ccp_wantoptions[f->unit];
    all_rejected[f->unit] = 0;

```

```

#ifdef MPPE
    if (go->mppe) {
        ccp_options *ao = &ccp_allowoptions[f->unit];
        int auth_mschap_bits = auth_done[f->unit];
        int numbits;

        /*
         * Start with a basic sanity check: mschap[v2] auth must be in
         * exactly one direction. RFC 3079 says that the keys are
         * 'derived from the credentials of the peer that initiated the call',
         * however the PPP protocol doesn't have such a concept, and pppd
         * cannot get this info externally. Instead we do the best we can.
         * NB: If MPPE is required, all other compression opts are invalid.
         * So, we return right away if we can't do it.
         */

        /* Leave only the mschap auth bits set */
        auth_mschap_bits &= (CHAP_MS_WITHPEER | CHAP_MS_PEER |
            CHAP_MS2_WITHPEER | CHAP_MS2_PEER);
        /* Count the mschap auths */
        auth_mschap_bits >>= CHAP_MS_SHIFT;
        numbits = 0;
        do {
            numbits += auth_mschap_bits & 1;
            auth_mschap_bits >>= 1;
        } while (auth_mschap_bits);
        if (numbits > 1) {
            error("MPPE required, but auth done in both directions.");
            lcp_close(f->unit, "MPPE required but not available");
            return;
        }
        if (!numbits) {
            error("MPPE required, but MS-CHAP[v2] auth not performed.");
            lcp_close(f->unit, "MPPE required but not available");
            return;
        }

        /* A plugin (eg radius) may not have obtained key material. */
        if (!mppe_keys_set) {
            error("MPPE required, but keys are not available. "
                "Possible plugin problem?");
            lcp_close(f->unit, "MPPE required but not available");
            return;
        }

        /* LM auth not supported for MPPE */
        if (auth_done[f->unit] & (CHAP_MS_WITHPEER | CHAP_MS_PEER)) {
            /* This might be noise */
            if (go->mppe & MPPE_OPT_40) {
                notice("Disabling 40-bit MPPE; MS-CHAP LM not supported");
                go->mppe &= ~MPPE_OPT_40;
                ccp_wantoptions[f->unit].mppe &= ~MPPE_OPT_40;
            }
        }
    }
}

```

```

/* Last check: can we actually negotiate something? */
if (!(go->mppe & (MPPE_OPT_40 | MPPE_OPT_128))) {
    /* Could be misconfig, could be 40-bit disabled above. */
    error("MPPE required, but both 40-bit and 128-bit disabled.");
    lcp_close(f->unit, "MPPE required but not available");
    return;
}

/* sync options */
ao->mppe = go->mppe;
/* MPPE is not compatible with other compression types */
ao->bsd_compress = go->bsd_compress = 0;
ao->predictor_1 = go->predictor_1 = 0;
ao->predictor_2 = go->predictor_2 = 0;
ao->deflate = go->deflate = 0;
}
#endif /* MPPE */

/*
 * Check whether the kernel knows about the various
 * compression methods we might request.
 */
#ifdef MPPE
    if (go->mppe) {
        opt_buf[0] = CI_MPPE;
        opt_buf[1] = CILEN_MPPE;
        MPPE_OPTS_TO_CI(go->mppe, &opt_buf[2]);
        /* Key material unimportant here. */
        if (ccp_test(f->unit, opt_buf, CILEN_MPPE + MPPE_MAX_KEY_LEN, 0) <= 0) {
            error("MPPE required, but kernel has no support.");
            lcp_close(f->unit, "MPPE required but not available");
        }
    }
#endif
    if (go->bsd_compress) {
        opt_buf[0] = CI_BSD_COMPRESS;
        opt_buf[1] = CILEN_BSD_COMPRESS;
        opt_buf[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION, BSD_MIN_BITS);
        if (ccp_test(f->unit, opt_buf, CILEN_BSD_COMPRESS, 0) <= 0)
            go->bsd_compress = 0;
    }
    if (go->deflate) {
        if (go->deflate_correct) {
            opt_buf[0] = CI_DEFLATE;
            opt_buf[1] = CILEN_DEFLATE;
            opt_buf[2] = DEFLATE_MAKE_OPT(DEFLATE_MIN_WORKS);
            opt_buf[3] = DEFLATE_CHK_SEQUENCE;
            if (ccp_test(f->unit, opt_buf, CILEN_DEFLATE, 0) <= 0)
                go->deflate_correct = 0;
        }
        if (go->deflate_draft) {
            opt_buf[0] = CI_DEFLATE_DRAFT;
            opt_buf[1] = CILEN_DEFLATE;
            opt_buf[2] = DEFLATE_MAKE_OPT(DEFLATE_MIN_WORKS);
        }
    }

```

```

    opt_buf[3] = DEFLATE_CHK_SEQUENCE;
    if (ccp_test(f->unit, opt_buf, CILEN_DEFLATE, 0) <= 0)
go->deflate_draft = 0;
}
if (!go->deflate_correct && !go->deflate_draft)
    go->deflate = 0;
}
if (go->predictor_1) {
opt_buf[0] = CI_PREDICTOR_1;
opt_buf[1] = CILEN_PREDICTOR_1;
if (ccp_test(f->unit, opt_buf, CILEN_PREDICTOR_1, 0) <= 0)
    go->predictor_1 = 0;
}
if (go->predictor_2) {
opt_buf[0] = CI_PREDICTOR_2;
opt_buf[1] = CILEN_PREDICTOR_2;
if (ccp_test(f->unit, opt_buf, CILEN_PREDICTOR_2, 0) <= 0)
    go->predictor_2 = 0;
}
}

/*
 * ccp_cilen - Return total length of our configuration info.
 */
static int
ccp_cilen(f)
    fsm *f;
{
    ccp_options *go = &ccp_gotoptions[f->unit];

    return (go->bsd_compress? CILEN_BSD_COMPRESS: 0)
+ (go->deflate? CILEN_DEFLATE: 0)
+ (go->predictor_1? CILEN_PREDICTOR_1: 0)
+ (go->predictor_2? CILEN_PREDICTOR_2: 0)
+ (go->mppe? CILEN_MPPE: 0);
}

/*
 * ccp_addci - put our requests in a packet.
 */
static void
ccp_addci(f, p, lenp)
    fsm *f;
    u_char *p;
    int *lenp;
{
    int res;
    ccp_options *go = &ccp_gotoptions[f->unit];
    u_char *p0 = p;

    /*
     * Add the compression types that we can receive, in decreasing
     * preference order. Get the kernel to allocate the first one
     * in case it gets Acked.
     */
}

```

```

#ifdef MPPE
    if (go->mppe) {
        u_char opt_buf[CILEN_MPPE + MPPE_MAX_KEY_LEN];

        p[0] = opt_buf[0] = CI_MPPE;
        p[1] = opt_buf[1] = CILEN_MPPE;
        MPPE_OPTS_TO_CI(go->mppe, &p[2]);
        MPPE_OPTS_TO_CI(go->mppe, &opt_buf[2]);
        BCOPY(mppe_recv_key, &opt_buf[CILEN_MPPE], MPPE_MAX_KEY_LEN);
        res = ccp_test(f->unit, opt_buf, CILEN_MPPE + MPPE_MAX_KEY_LEN, 0);
        if (res > 0)
            p += CILEN_MPPE;
        else
            /* This shouldn't happen, we've already tested it! */
            lcp_close(f->unit, "MPPE required but not available in kernel");
    }
#endif
    if (go->deflate) {
        p[0] = go->deflate_correct? CI_DEFLATE: CI_DEFLATE_DRAFT;
        p[1] = CILEN_DEFLATE;
        p[2] = DEFLATE_MAKE_OPT(go->deflate_size);
        p[3] = DEFLATE_CHK_SEQUENCE;
        if (p != p0) {
            p += CILEN_DEFLATE;
        } else {
            for (;;) {
                if (go->deflate_size < DEFLATE_MIN_WORKS) {
                    go->deflate = 0;
                    break;
                }
                res = ccp_test(f->unit, p, CILEN_DEFLATE, 0);
                if (res > 0) {
                    p += CILEN_DEFLATE;
                    break;
                } else if (res < 0) {
                    go->deflate = 0;
                    break;
                }
                --go->deflate_size;
                p[2] = DEFLATE_MAKE_OPT(go->deflate_size);
            }
        }
        if (p != p0 && go->deflate_correct && go->deflate_draft) {
            p[0] = CI_DEFLATE_DRAFT;
            p[1] = CILEN_DEFLATE;
            p[2] = p[2] - CILEN_DEFLATE;
            p[3] = DEFLATE_CHK_SEQUENCE;
            p += CILEN_DEFLATE;
        }
    }
    if (go->bsd_compress) {
        p[0] = CI_BSD_COMPRESS;
        p[1] = CILEN_BSD_COMPRESS;
        p[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION, go->bsd_bits);
    }

```

```

if (p != p0) {
    p += CILEN_BSD_COMPRESS; /* not the first option */
} else {
    for (;;) {
if (go->bsd_bits < BSD_MIN_BITS) {
    go->bsd_compress = 0;
    break;
}
res = ccp_test(f->unit, p, CILEN_BSD_COMPRESS, 0);
if (res > 0) {
    p += CILEN_BSD_COMPRESS;
    break;
} else if (res < 0) {
    go->bsd_compress = 0;
    break;
}
--go->bsd_bits;
p[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION, go->bsd_bits);
    }
}
    }
    /* XXX Should Predictor 2 be preferable to Predictor 1? */
    if (go->predictor_1) {
p[0] = CI_PREDICTOR_1;
p[1] = CILEN_PREDICTOR_1;
if (p == p0 && ccp_test(f->unit, p, CILEN_PREDICTOR_1, 0) <= 0) {
    go->predictor_1 = 0;
} else {
    p += CILEN_PREDICTOR_1;
}
    }
    if (go->predictor_2) {
p[0] = CI_PREDICTOR_2;
p[1] = CILEN_PREDICTOR_2;
if (p == p0 && ccp_test(f->unit, p, CILEN_PREDICTOR_2, 0) <= 0) {
    go->predictor_2 = 0;
} else {
    p += CILEN_PREDICTOR_2;
}
    }

    go->method = (p > p0)? p0[0]: -1;

    *lenp = p - p0;
}

/*
 * ccp_ackci - process a received configure-ack, and return
 * 1 iff the packet was OK.
 */
static int
ccp_ackci(f, p, len)
    fsm *f;
    u_char *p;
    int len;

```



```

{
    ccp_options *go = &ccp_gotoptions[f->unit];
    u_char *p0 = p;

#ifdef MPPE
    if (go->mppe) {
        u_char opt_buf[CILEN_MPPE];

        opt_buf[0] = CI_MPPE;
        opt_buf[1] = CILEN_MPPE;
        MPPE_OPTS_TO_CI(go->mppe, &opt_buf[2]);
        if (len < CILEN_MPPE || memcmp(opt_buf, p, CILEN_MPPE))
            return 0;
        p += CILEN_MPPE;
        len -= CILEN_MPPE;
        /* XXX Cope with first/fast ack */
        if (len == 0)
            return 1;
    }
#endif

    if (go->deflate) {
        if (len < CILEN_DEFLATE
            || p[0] != (go->deflate_correct? CI_DEFLATE: CI_DEFLATE_DRAFT)
            || p[1] != CILEN_DEFLATE
            || p[2] != DEFLATE_MAKE_OPT(go->deflate_size)
            || p[3] != DEFLATE_CHK_SEQUENCE)
            return 0;
        p += CILEN_DEFLATE;
        len -= CILEN_DEFLATE;
        /* XXX Cope with first/fast ack */
        if (len == 0)
            return 1;
        if (go->deflate_correct && go->deflate_draft) {
            if (len < CILEN_DEFLATE
                || p[0] != CI_DEFLATE_DRAFT
                || p[1] != CILEN_DEFLATE
                || p[2] != DEFLATE_MAKE_OPT(go->deflate_size)
                || p[3] != DEFLATE_CHK_SEQUENCE)
                return 0;
            p += CILEN_DEFLATE;
            len -= CILEN_DEFLATE;
        }
    }

    if (go->bsd_compress) {
        if (len < CILEN_BSD_COMPRESS
            || p[0] != CI_BSD_COMPRESS || p[1] != CILEN_BSD_COMPRESS
            || p[2] != BSD_MAKE_OPT(BSD_CURRENT_VERSION, go->bsd_bits))
            return 0;
        p += CILEN_BSD_COMPRESS;
        len -= CILEN_BSD_COMPRESS;
        /* XXX Cope with first/fast ack */
        if (p == p0 && len == 0)
            return 1;
    }
}

```

```

    if (go->predictor_1) {
    if (len < CILEN_PREDICTOR_1
        || p[0] != CI_PREDICTOR_1 || p[1] != CILEN_PREDICTOR_1)
        return 0;
    p += CILEN_PREDICTOR_1;
    len -= CILEN_PREDICTOR_1;
    /* XXX Cope with first/fast ack */
    if (p == p0 && len == 0)
        return 1;
    }
    if (go->predictor_2) {
    if (len < CILEN_PREDICTOR_2
        || p[0] != CI_PREDICTOR_2 || p[1] != CILEN_PREDICTOR_2)
        return 0;
    p += CILEN_PREDICTOR_2;
    len -= CILEN_PREDICTOR_2;
    /* XXX Cope with first/fast ack */
    if (p == p0 && len == 0)
        return 1;
    }

    if (len != 0)
    return 0;
    return 1;
}

/*
 * ccp_nakci - process received configure-nak.
 * Returns 1 iff the nak was OK.
 */
static int
ccp_nakci(f, p, len, treat_as_reject)
    fsm *f;
    u_char *p;
    int len;
    int treat_as_reject;
{
    ccp_options *go = &ccp_gotoptions[f->unit];
    ccp_options no; /* options we've seen already */
    ccp_options try; /* options to ask for next time */

    memset(&no, 0, sizeof(no));
    try = *go;

#ifdef MPPE
    if (go->mppe && len >= CILEN_MPPE
        && p[0] == CI_MPPE && p[1] == CILEN_MPPE) {
        no.mppe = 1;
        /*
         * Peer wants us to use a different strength or other setting.
         * Fail if we aren't willing to use his suggestion.
         */
        MPPE_CI_TO_OPTS(&p[2], try.mppe);
        if ((try.mppe & MPPE_OPT_STATEFUL) && refuse_mppe_stateful) {
            error("Refusing MPPE stateful mode offered by peer");

```

```

    try.mppe = 0;
} else if (((go->mppe | MPPE_OPT_STATEFUL) & try.mppe) != try.mppe) {
    /* Peer must have set options we didn't request (suggest) */
    try.mppe = 0;
}

if (!try.mppe) {
    error("MPPE required but peer negotiation failed");
    lcp_close(f->unit, "MPPE required but peer negotiation failed");
}
}

#endif /* MPPE */

if (go->deflate && len >= CILEN_DEFLATE
&& p[0] == (go->deflate_correct? CI_DEFLATE: CI_DEFLATE_DRAFT)
&& p[1] == CILEN_DEFLATE) {
    no.deflate = 1;
    /*
     * Peer wants us to use a different code size or something.
     * Stop asking for Deflate if we don't understand his suggestion.
     */
    if (DEFLATE_METHOD(p[2]) != DEFLATE_METHOD_VAL
        || DEFLATE_SIZE(p[2]) < DEFLATE_MIN_WORKS
        || p[3] != DEFLATE_CHK_SEQUENCE)
        try.deflate = 0;
    else if (DEFLATE_SIZE(p[2]) < go->deflate_size)
        try.deflate_size = DEFLATE_SIZE(p[2]);
    p += CILEN_DEFLATE;
    len -= CILEN_DEFLATE;
    if (go->deflate_correct && go->deflate_draft
        && len >= CILEN_DEFLATE && p[0] == CI_DEFLATE_DRAFT
        && p[1] == CILEN_DEFLATE) {
        p += CILEN_DEFLATE;
        len -= CILEN_DEFLATE;
    }
}

if (go->bsd_compress && len >= CILEN_BSD_COMPRESS
&& p[0] == CI_BSD_COMPRESS && p[1] == CILEN_BSD_COMPRESS) {
    no.bsd_compress = 1;
    /*
     * Peer wants us to use a different number of bits
     * or a different version.
     */
    if (BSD_VERSION(p[2]) != BSD_CURRENT_VERSION)
        try.bsd_compress = 0;
    else if (BSD_NBITS(p[2]) < go->bsd_bits)
        try.bsd_bits = BSD_NBITS(p[2]);
    p += CILEN_BSD_COMPRESS;
    len -= CILEN_BSD_COMPRESS;
}

/*
 * Predictor-1 and 2 have no options, so they can't be Naked.
 */

```

```

    * There may be remaining options but we ignore them.
    */

    if (f->state != OPENED)
    *go = try;
    return 1;
}

/*
 * ccp_rejci - reject some of our suggested compression methods.
 */
static int
ccp_rejci(f, p, len)
    fsm *f;
    u_char *p;
    int len;
{
    ccp_options *go = &ccp_gotoptions[f->unit];
    ccp_options try; /* options to request next time */

    try = *go;

    /*
     * Cope with empty configure-rejects by ceasing to send
     * configure-requests.
     */
    if (len == 0 && all_rejected[f->unit])
    return -1;

#ifdef MPPE
    if (go->mppe && len >= CILEN_MPPE
    && p[0] == CI_MPPE && p[1] == CILEN_MPPE) {
        error("MPPE required but peer refused");
        lcp_close(f->unit, "MPPE required but peer refused");
        p += CILEN_MPPE;
        len -= CILEN_MPPE;
    }
#endif
    if (go->deflate_correct && len >= CILEN_DEFLATE
    && p[0] == CI_DEFLATE && p[1] == CILEN_DEFLATE) {
    if (p[2] != DEFLATE_MAKE_OPT(go->deflate_size)
    || p[3] != DEFLATE_CHK_SEQUENCE)
        return 0; /* Rej is bad */
    try.deflate_correct = 0;
    p += CILEN_DEFLATE;
    len -= CILEN_DEFLATE;
    }
    if (go->deflate_draft && len >= CILEN_DEFLATE
    && p[0] == CI_DEFLATE_DRAFT && p[1] == CILEN_DEFLATE) {
    if (p[2] != DEFLATE_MAKE_OPT(go->deflate_size)
    || p[3] != DEFLATE_CHK_SEQUENCE)
        return 0; /* Rej is bad */
    try.deflate_draft = 0;
    p += CILEN_DEFLATE;
    len -= CILEN_DEFLATE;
    }

```

```

    }
    if (!try.deflate_correct && !try.deflate_draft)
try.deflate = 0;
    if (go->bsd_compress && len >= CILEN_BSD_COMPRESS
&& p[0] == CI_BSD_COMPRESS && p[1] == CILEN_BSD_COMPRESS) {
if (p[2] != BSD_MAKE_OPT(BSD_CURRENT_VERSION, go->bsd_bits))
    return 0;
try.bsd_compress = 0;
p += CILEN_BSD_COMPRESS;
len -= CILEN_BSD_COMPRESS;
    }
    if (go->predictor_1 && len >= CILEN_PREDICTOR_1
&& p[0] == CI_PREDICTOR_1 && p[1] == CILEN_PREDICTOR_1) {
try.predictor_1 = 0;
p += CILEN_PREDICTOR_1;
len -= CILEN_PREDICTOR_1;
    }
    if (go->predictor_2 && len >= CILEN_PREDICTOR_2
&& p[0] == CI_PREDICTOR_2 && p[1] == CILEN_PREDICTOR_2) {
try.predictor_2 = 0;
p += CILEN_PREDICTOR_2;
len -= CILEN_PREDICTOR_2;
    }

    if (len != 0)
return 0;

    if (f->state != OPENED)
*go = try;

    return 1;
}

/*
 * ccp_reqci - processed a received configure-request.
 * Returns CONFACK, CONFNAK or CONFREJ and the packet modified
 * appropriately.
 */
static int
ccp_reqci(f, p, lenp, dont_nak)
    fsm *f;
    u_char *p;
    int *lenp;
    int dont_nak;
{
    int ret, newret, res;
    u_char *p0, *retp;
    int len, clen, type, nb;
    ccp_options *ho = &ccp_hisoptions[f->unit];
    ccp_options *ao = &ccp_allowoptions[f->unit];
#ifdef MPPE
    bool rej_for_ci_mppe = 1; /* Are we rejecting based on a bad/missing */
/* CI_MPPE, or due to other options? */
#endif
}

```

```

    ret = CONFACK;
    retp = p0 = p;
    len = *lenp;

    memset(ho, 0, sizeof(ccp_options));
    ho->method = (len > 0)? p[0]: -1;

    while (len > 0) {
newret = CONFACK;
    if (len < 2 || p[1] < 2 || p[1] > len) {
        /* length is bad */
        clen = len;
        newret = CONFREJ;
    } else {
        type = p[0];
        clen = p[1];

        switch (type) {
#ifdef MPPE
        case CI_MPPE:
            if (!ao->mppe || clen != CILEN_MPPE) {
                newret = CONFREJ;
                break;
            }
            MPPE_CI_TO_OPTS(&p[2], ho->mppe);

            /* Nak if anything unsupported or unknown are set. */
            if (ho->mppe & MPPE_OPT_UNSUPPORTED) {
                newret = CONFNAK;
                ho->mppe &= ~MPPE_OPT_UNSUPPORTED;
            }
            if (ho->mppe & MPPE_OPT_UNKNOWN) {
                newret = CONFNAK;
                ho->mppe &= ~MPPE_OPT_UNKNOWN;
            }

            /* Check state opt */
            if (ho->mppe & MPPE_OPT_STATEFUL) {
                /*
                 * We can Nak and request stateless, but it's a
                 * lot easier to just assume the peer will request
                 * it if he can do it; stateful mode is bad over
                 * the Internet -- which is where we expect MPPE.
                 */
                if (refuse_mppe_stateful) {
                    error("Refusing MPPE stateful mode offered by peer");
                    newret = CONFREJ;
                    break;
                }
            }

            /* Find out which of {S,L} are set. */
            if ((ho->mppe & MPPE_OPT_128)

```

```

    && (ho->mppe & MPPE_OPT_40)) {
/* Both are set, negotiate the strongest. */
newret = CONFNAK;
if (ao->mppe & MPPE_OPT_128)
ho->mppe &= ~MPPE_OPT_40;
else if (ao->mppe & MPPE_OPT_40)
ho->mppe &= ~MPPE_OPT_128;
else {
newret = CONFREJ;
break;
}
} else if (ho->mppe & MPPE_OPT_128) {
if (!(ao->mppe & MPPE_OPT_128)) {
newret = CONFREJ;
break;
}
} else if (ho->mppe & MPPE_OPT_40) {
if (!(ao->mppe & MPPE_OPT_40)) {
newret = CONFREJ;
break;
}
} else {
/* Neither are set. */
/* We cannot accept this. */
newret = CONFNAK;
/* Give the peer our idea of what can be used,
so it can choose and confirm */
ho->mppe = ao->mppe;
}

/* rebuild the opts */
MPPE_OPTS_TO_CI(ho->mppe, &p[2]);
if (newret == CONFACK) {
u_char opt_buf[CILEN_MPPE + MPPE_MAX_KEY_LEN];
int mtu;

BCOPY(p, opt_buf, CILEN_MPPE);
BCOPY(mppe_send_key, &opt_buf[CILEN_MPPE],
MPPE_MAX_KEY_LEN);
if (ccp_test(f->unit, opt_buf,
CILEN_MPPE + MPPE_MAX_KEY_LEN, 1) <= 0) {
/* This shouldn't happen, we've already tested it! */
error("MPPE required, but kernel has no support.");
lcp_close(f->unit, "MPPE required but not available");
newret = CONFREJ;
break;
}
/*
* We need to decrease the interface MTU by MPPE_PAD
* because MPPE frames **grow**. The kernel [must]
* allocate MPPE_PAD extra bytes in xmit buffers.
*/
mtu = netif_get_mtu(f->unit);
if (mtu)

```

```

netif_set_mtu(f->unit, mtu - MPPE_PAD);
    else
newret = CONFREJ;
}

/*
 * We have accepted MPPE or are willing to negotiate
 * MPPE parameters. A CONFREJ is due to subsequent
 * (non-MPPE) processing.
 */
rej_for_ci_mppe = 0;
break;
#endif /* MPPE */
    case CI_DEFLATE:
    case CI_DEFLATE_DRAFT:
if (!ao->deflate || clen != CILEN_DEFLATE
    || (!ao->deflate_correct && type == CI_DEFLATE)
    || (!ao->deflate_draft && type == CI_DEFLATE_DRAFT)) {
    newret = CONFREJ;
    break;
}

ho->deflate = 1;
ho->deflate_size = nb = DEFLATE_SIZE(p[2]);
if (DEFLATE_METHOD(p[2]) != DEFLATE_METHOD_VAL
    || p[3] != DEFLATE_CHK_SEQUENCE
    || nb > ao->deflate_size || nb < DEFLATE_MIN_WORKS) {
    newret = CONFNAK;
    if (!dont_nak) {
p[2] = DEFLATE_MAKE_OPT(ao->deflate_size);
p[3] = DEFLATE_CHK_SEQUENCE;
/* fall through to test this #bits below */
    } else
break;
}

/*
 * Check whether we can do Deflate with the window
 * size they want. If the window is too big, reduce
 * it until the kernel can cope and nak with that.
 * We only check this for the first option.
 */
if (p == p0) {
    for (;;) {
res = ccp_test(f->unit, p, CILEN_DEFLATE, 1);
if (res > 0)
    break; /* it's OK now */
if (res < 0 || nb == DEFLATE_MIN_WORKS || dont_nak) {
    newret = CONFREJ;
    p[2] = DEFLATE_MAKE_OPT(ho->deflate_size);
    break;
}
newret = CONFNAK;
--nb;
p[2] = DEFLATE_MAKE_OPT(nb);

```



```

    }
}
break;

    case CI_BSD_COMPRESS:
if (!ao->bsd_compress || clen != CILEN_BSD_COMPRESS) {
    newret = CONFREJ;
    break;
}

ho->bsd_compress = 1;
ho->bsd_bits = nb = BSD_NBITS(p[2]);
if (BSD_VERSION(p[2]) != BSD_CURRENT_VERSION
    || nb > ao->bsd_bits || nb < BSD_MIN_BITS) {
    newret = CONFNAK;
    if (!dont_nak) {
p[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION, ao->bsd_bits);
/* fall through to test this #bits below */
    } else
break;
}

/*
 * Check whether we can do BSD-Compress with the code
 * size they want. If the code size is too big, reduce
 * it until the kernel can cope and nak with that.
 * We only check this for the first option.
 */
if (p == p0) {
    for (;;) {
res = ccp_test(f->unit, p, CILEN_BSD_COMPRESS, 1);
if (res > 0)
    break;
if (res < 0 || nb == BSD_MIN_BITS || dont_nak) {
    newret = CONFREJ;
    p[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION,
ho->bsd_bits);
    break;
}
newret = CONFNAK;
--nb;
p[2] = BSD_MAKE_OPT(BSD_CURRENT_VERSION, nb);
    }
}
break;

    case CI_PREDICTOR_1:
if (!ao->predictor_1 || clen != CILEN_PREDICTOR_1) {
    newret = CONFREJ;
    break;
}

ho->predictor_1 = 1;
if (p == p0

```

```

    && ccp_test(f->unit, p, CILEN_PREDICTOR_1, 1) <= 0) {
        newret = CONFREJ;
    }
    break;

    case CI_PREDICTOR_2:
    if (!ao->predictor_2 || clen != CILEN_PREDICTOR_2) {
        newret = CONFREJ;
        break;
    }

    ho->predictor_2 = 1;
    if (p == p0
        && ccp_test(f->unit, p, CILEN_PREDICTOR_2, 1) <= 0) {
        newret = CONFREJ;
    }
    break;

    default:
    newret = CONFREJ;
    }
}

if (newret == CONFNAK && dont_nak)
    newret = CONFREJ;
if (!(newret == CONFACK || (newret == CONFNAK && ret == CONFREJ))) {
    /* we're returning this option */
    if (newret == CONFREJ && ret == CONFNAK)
    retp = p0;
    ret = newret;
    if (p != retp)
    BCOPY(p, retp, clen);
    retp += clen;
}

p += clen;
len -= clen;
}

if (ret != CONFACK) {
if (ret == CONFREJ && *lenp == retp - p0)
    all_rejected[f->unit] = 1;
else
    *lenp = retp - p0;
}
#ifdef MPPE
    if (ret == CONFREJ && ao->mppe && rej_for_ci_mppe) {
        error("MPPE required but peer negotiation failed");
        lcp_close(f->unit, "MPPE required but peer negotiation failed");
    }
#endif
    return ret;
}

/*

```

```

    * Make a string name for a compression method (or 2).
    */
static char *
method_name(opt, opt2)
    ccp_options *opt, *opt2;
{
    static char result[64];

    if (!ANY_COMPRESS(*opt))
return "(none)";
    switch (opt->method) {
#ifdef MPPE
        case CI_MPPE:
        {
            char *p = result;
            char *q = result + sizeof(result); /* 1 past result */

            slprintf(p, q - p, "MPPE ");
            p += 5;
            if (opt->mppe & MPPE_OPT_128) {
                slprintf(p, q - p, "128-bit ");
                p += 8;
            }
            if (opt->mppe & MPPE_OPT_40) {
                slprintf(p, q - p, "40-bit ");
                p += 7;
            }
            if (opt->mppe & MPPE_OPT_STATEFUL)
                slprintf(p, q - p, "stateful");
            else
                slprintf(p, q - p, "stateless");

            break;
        }
    #endif
        case CI_DEFLATE:
        case CI_DEFLATE_DRAFT:
            if (opt2 != NULL && opt2->deflate_size != opt->deflate_size)
                slprintf(result, sizeof(result), "Deflate%s (%d/%d)",
                    (opt->method == CI_DEFLATE_DRAFT? "(old#)": ""),
                    opt->deflate_size, opt2->deflate_size);
            else
                slprintf(result, sizeof(result), "Deflate%s (%d)",
                    (opt->method == CI_DEFLATE_DRAFT? "(old#)": ""),
                    opt->deflate_size);
            break;
        case CI_BSD_COMPRESS:
            if (opt2 != NULL && opt2->bsd_bits != opt->bsd_bits)
                slprintf(result, sizeof(result), "BSD-Compress (%d/%d)",
                    opt->bsd_bits, opt2->bsd_bits);
            else
                slprintf(result, sizeof(result), "BSD-Compress (%d)",
                    opt->bsd_bits);
            break;
    }
}

```

```

        case CI_PREDICTOR_1:
return "Predictor 1";
        case CI_PREDICTOR_2:
return "Predictor 2";
        default:
sprintf(result, sizeof(result), "Method %d", opt->method);
    }
    return result;
}

/*
 * CCP has come up - inform the kernel driver and log a message.
 */
static void
ccp_up(f)
    fsm *f;
{
    ccp_options *go = &ccp_gotoptions[f->unit];
    ccp_options *ho = &ccp_hisoptions[f->unit];
    char method1[64];

    ccp_flags_set(f->unit, 1, 1);
    if (ANY_COMPRESS(*go)) {
    if (ANY_COMPRESS(*ho)) {
        if (go->method == ho->method) {
notice("%s compression enabled", method_name(go, ho));
        } else {
strcpy(method1, method_name(go, NULL), sizeof(method1));
notice("%s / %s compression enabled",
        method1, method_name(ho, NULL));
        }
    } else
        notice("%s receive compression enabled", method_name(go, NULL));
    } else if (ANY_COMPRESS(*ho))
notice("%s transmit compression enabled", method_name(ho, NULL));
#ifdef MPPE
    if (go->mppe) {
BZERO(mppe_rcv_key, MPPE_MAX_KEY_LEN);
BZERO(mppe_snd_key, MPPE_MAX_KEY_LEN);
continue_networks(f->unit);/* Bring up IP et al */
    }
#endif
}

/*
 * CCP has gone down - inform the kernel driver.
 */
static void
ccp_down(f)
    fsm *f;
{
    if (ccp_localstate[f->unit] & RACK_PENDING)
UNTIMEOUT(ccp_rack_timeout, f);
    ccp_localstate[f->unit] = 0;
    ccp_flags_set(f->unit, 1, 0);
}

```

```

#ifdef MPPE
    if (ccp_gotoptions[f->unit].mppe) {
ccp_gotoptions[f->unit].mppe = 0;
if (lcp_fsm[f->unit].state == OPENED) {
    /* If LCP is not already going down, make sure it does. */
    error("MPPE disabled");
    lcp_close(f->unit, "MPPE disabled");
}
}
#endif

/*
 * Print the contents of a CCP packet.
 */
static char *ccp_codenames[] = {
    "ConfReq", "ConfAck", "ConfNak", "ConfRej",
    "TermReq", "TermAck", "CodeRej",
    NULL, NULL, NULL, NULL, NULL, NULL,
    "ResetReq", "ResetAck",
};

static int
ccp_printpkt(p, plen, printer, arg)
    u_char *p;
    int plen;
    void (*printer) __P((void *, char *, ...));
    void *arg;
{
    u_char *p0, *optend;
    int code, id, len;
    int optlen;

    p0 = p;
    if (plen < HEADERLEN)
return 0;
    code = p[0];
    id = p[1];
    len = (p[2] << 8) + p[3];
    if (len < HEADERLEN || len > plen)
return 0;

    if (code >= 1 && code <= sizeof(ccp_codenames) / sizeof(char *)
&& ccp_codenames[code-1] != NULL)
printer(arg, " %s", ccp_codenames[code-1]);
    else
printer(arg, " code=0x%x", code);
    printer(arg, " id=0x%x", id);
    len -= HEADERLEN;
    p += HEADERLEN;

    switch (code) {
case CONFREQ:
case CONFACK:

```

```

        case CONFNAK:
        case CONFREJ:
/* print list of possible compression methods */
while (len >= 2) {
    code = p[0];
    optlen = p[1];
    if (optlen < 2 || optlen > len)
break;
    printer(arg, " <");
    len -= optlen;
    optend = p + optlen;
    switch (code) {
#ifdef MPPE
        case CI_MPPE:
if (optlen >= CILEN_MPPE) {
    u_char mppe_opts;

    MPPE_CI_TO_OPTS(&p[2], mppe_opts);
    printer(arg, "mppe %s %s %s %s %s %s",
(p[2] & MPPE_H_BIT)? "+H": "-H",
(p[5] & MPPE_M_BIT)? "+M": "-M",
(p[5] & MPPE_S_BIT)? "+S": "-S",
(p[5] & MPPE_L_BIT)? "+L": "-L",
(p[5] & MPPE_D_BIT)? "+D": "-D",
(p[5] & MPPE_C_BIT)? "+C": "-C",
(mppe_opts & MPPE_OPT_UNKNOWN)? "+U": "");
    if (mppe_opts & MPPE_OPT_UNKNOWN)
printer(arg, " (%.2x %.2x %.2x %.2x)",
p[2], p[3], p[4], p[5]);
    p += CILEN_MPPE;
}
break;
#endif
        case CI_DEFLATE:
        case CI_DEFLATE_DRAFT:
if (optlen >= CILEN_DEFLATE) {
    printer(arg, "deflate%s %d",
(code == CI_DEFLATE_DRAFT? "(old#)": ""),
DEFLATE_SIZE(p[2]));
    if (DEFLATE_METHOD(p[2]) != DEFLATE_METHOD_VAL)
printer(arg, " method %d", DEFLATE_METHOD(p[2]));
    if (p[3] != DEFLATE_CHK_SEQUENCE)
printer(arg, " check %d", p[3]);
    p += CILEN_DEFLATE;
}
break;
        case CI_BSD_COMPRESS:
if (optlen >= CILEN_BSD_COMPRESS) {
    printer(arg, "bsd v%d %d", BSD_VERSION(p[2]),
BSD_NBITS(p[2]));
    p += CILEN_BSD_COMPRESS;
}
break;
        case CI_PREDICTOR_1:
if (optlen >= CILEN_PREDICTOR_1) {

```

```

        printer(arg, "predictor 1");
        p += CILEN_PREDICTOR_1;
    }
    break;
    case CI_PREDICTOR_2:
    if (optlen >= CILEN_PREDICTOR_2) {
        printer(arg, "predictor 2");
        p += CILEN_PREDICTOR_2;
    }
    break;
    }
    while (p < optend)
    printer(arg, "%.2x", *p++);
    printer(arg, ">");
}
break;

    case TERMACK:
    case TERMREQ:
    if (len > 0 && *p >= ' ' && *p < 0x7f) {
        print_string((char *)p, len, printer, arg);
        p += len;
        len = 0;
    }
    break;
    }

    /* dump out the rest of the packet in hex */
    while (--len >= 0)
    printer(arg, "%.2x", *p++);

    return p - p0;
}

/*
 * We have received a packet that the decompressor failed to
 * decompress. Here we would expect to issue a reset-request, but
 * Motorola has a patent on resetting the compressor as a result of
 * detecting an error in the decompressed data after decompression.
 * (See US patent 5,130,993; international patent publication number
 * WO 91/10289; Australian patent 73296/91.)
 *
 * So we ask the kernel whether the error was detected after
 * decompression; if it was, we take CCP down, thus disabling
 * compression :-), otherwise we issue the reset-request.
 */
static void
ccp_datainput(unit, pkt, len)
    int unit;
    u_char *pkt;
    int len;
{
    fsm *f;

```

```

    f = &ccp_fsm[unit];
    if (f->state == OPENED) {
    if (ccp_fatal_error(unit)) {
    /*
    * Disable compression by taking CCP down.
    */
    error("Lost compression sync: disabling compression");
    ccp_close(unit, "Lost compression sync");
#ifdef MPPE
    /*
    * If we were doing MPPE, we must also take the link down.
    */
    if (ccp_gotoptions[unit].mppe) {
    error("Too many MPPE errors, closing LCP");
    lcp_close(unit, "Too many MPPE errors");
    }
#endif
    } else {
    /*
    * Send a reset-request to reset the peer's compressor.
    * We don't do that if we are still waiting for an
    * acknowledgement to a previous reset-request.
    */
    if (!(ccp_localstate[f->unit] & RACK_PENDING)) {
    fsm_sdata(f, CCP_RESETREQ, f->reqid = ++f->id, NULL, 0);
    TIMEOUT(ccp_rack_timeout, f, RACKTIMEOUT);
    ccp_localstate[f->unit] |= RACK_PENDING;
    } else
    ccp_localstate[f->unit] |= RREQ_REPEAT;
    }
    }

    /*
    * Timeout waiting for reset-ack.
    */
    static void
    ccp_rack_timeout(arg)
    void *arg;
    {
    fsm *f = arg;

    if (f->state == OPENED && ccp_localstate[f->unit] & RREQ_REPEAT) {
    fsm_sdata(f, CCP_RESETREQ, f->reqid, NULL, 0);
    TIMEOUT(ccp_rack_timeout, f, RACKTIMEOUT);
    ccp_localstate[f->unit] &= ~RREQ_REPEAT;
    } else
    ccp_localstate[f->unit] &= ~RACK_PENDING;
    }

    /*
    * Chat -- a program for automatic session establishment (i.e. dial
    * the phone and log in).
    *
    * Standard termination codes:

```


* 0 - successful completion of the script
* 1 - invalid argument, expect string too large, etc.
* 2 - error on an I/O operation or fatal error condition.
* 3 - timeout waiting for a simple string.
* 4 - the first string declared as "ABORT"
* 5 - the second string declared as "ABORT"
* 6 - ... and so on for successive ABORT strings.
*
*This software is in the public domain.
*
* -----
*22-May-99 added environment substitution, enabled with -E switch.
*Andreas Arens <andras@cityweb.de>.
*
*12-May-99 added a feature to read data to be sent from a file,
*if the send string starts with @. Idea from gpk <gpk@onramp.net>.
*
*added -T and -U option and \T and \U substitution to pass a phone
*number into chat script. Two are needed for some ISDN TA applications.
*Keith Dart <kdart@cisco.com>
*
*
*Added SAY keyword to send output to stderr.
* This allows to turn ECHO OFF and to output specific, user selected,
* text to give progress messages. This best works when stderr
* exists (i.e.: pppd in nodetach mode).
*
* Added HANGUP directives to allow for us to be called
* back. When HANGUP is set to NO, chat will not hangup at HUP signal.
* We rely on timeouts in that case.
*
* Added CLR_ABORT to clear previously set ABORT string. This has been
* dictated by the HANGUP above as "NO CARRIER" (for example) must be
* an ABORT condition until we know the other host is going to close
* the connection for call back. As soon as we have completed the
* first stage of the call back sequence, "NO CARRIER" is a valid, non
* fatal string. As soon as we got called back (probably get "CONNECT"),
* we should re-arm the ABORT "NO CARRIER". Hence the CLR_ABORT command.
* Note that CLR_ABORT packs the abort_strings[] array so that we do not
* have unused entries not being reclaimed.
*
* In the same vein as above, added CLR_REPORT keyword.
*
* Allow for comments. Line starting with '#' are comments and are
* ignored. If a '#' is to be expected as the first character, the
* expect string must be quoted.
*
*
*Francis Demierre <Francis@SwissMail.Com>
* Thu May 15 17:15:40 MET DST 1997
*
*
* Added -r "report file" switch & REPORT keyword.
* Robert Geer <bgeer@xmission.com>

```

*
*   Added -s "use stderr" and -S "don't use syslog" switches.
*       June 18, 1997
*       Karl O. Pinc <kop@meme.com>
*
*
*Added -e "echo" switch & ECHO keyword
*Dick Streefland <dicks@tasking.nl>
*
*
*Considerable updates and modifications by
*Al Longyear <longyear@pobox.com>
*Paul Mackerras <paulus@cs.anu.edu.au>
*
*
*The original author is:
*
*Karl Fox <karl@MorningStar.Com>
*Morning Star Technologies, Inc.
*1760 Zollinger Road
*Columbus, OH 43221
*(614)451-1883
*
*/

#ifndef __STDC__
#define const
#endif

#ifndef lint
static const char resid[] = "$Id: chat.c,v 1.30 2004/01/17 05:47:55 carlsonj Exp $";
#endif

#include <stdio.h>
#include <ctype.h>
#include <time.h>
#include <fcntl.h>
#include <signal.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <syslog.h>

#ifndef TERMIO
#undef TERMIOS
#define TERMIOS
#endif

#ifdef TERMIO
#include <termio.h>
#endif
#ifdef TERMIOS

```

```

#include <termios.h>
#endif

#define STR_LEN 1024

#ifndef SIGTYPE
#define SIGTYPE void
#endif

#undef __P
#undef __V

#ifdef __STDC__
#include <stdarg.h>
#define __V(x) x
#define __P(x) x
#else
#include <varargs.h>
#define __V(x)(va_alist) va_dcl
#define __P(x)()
#define const
#endif

#ifndef O_NONBLOCK
#define O_NONBLOCK O_NDELAY
#endif

#ifdef SUNOS
extern int sys_nerr;
extern char *sys_errlist[];
#define memmove(to, from, n) bcopy(from, to, n)
#define strerror(n)((unsigned)(n) < sys_nerr? sys_errlist[(n)] : \
    "unknown error")
#endif

/***** Micro getopt() *****/
#define OPTION(c,v) (_O & 2 && **v? (*v)++: !c || _O & 4? 0: (!(_O & 1) && \
    (--c, ++v), _O = 4, c && **v == '-' && v[0][1]? **v++ == '-' \
    && !v[0][1]? (--c, ++v, 0): (_O = 2, (*v)++): 0))
#define OPTARG(c,v) (_O & 2? **v || (++v, --c)? (_O = 1, --c, *v++): \
    (_O = 4, (char*)0): (char*)0)
#define OPTONLYARG(c,v) (_O & 2 && **v? (_O = 1, --c, *v++): (char*)0)
#define ARG(c,v) (c? (--c, *v++): (char*)0)

static int _O = 0; /* Internal state */
/***** Micro getopt() *****/

char *program_name;

#define MAX_ABORTS 50
#define MAX_REPORTS 50
#define DEFAULT_CHAT_TIMEOUT 45

int echo      = 0;

```

```

int verbose    = 0;
int to_log     = 1;
int to_stderr  = 0;
int Verbose    = 0;
int quiet      = 0;
int report     = 0;
int use_env    = 0;
int exit_code  = 0;
FILE* report_fp = (FILE *) 0;
char *report_file = (char *) 0;
char *chat_file  = (char *) 0;
char *phone_num  = (char *) 0;
char *phone_num2 = (char *) 0;
int timeout     = DEFAULT_CHAT_TIMEOUT;

int have_tty_parameters = 0;

#ifdef TERMIO
#define term_parms struct termio
#define get_term_param(param) ioctl(0, TCGETA, param)
#define set_term_param(param) ioctl(0, TCSETA, param)
struct termio saved_tty_parameters;
#endif

#ifdef TERMIOS
#define term_parms struct termios
#define get_term_param(param) tcgetattr(0, param)
#define set_term_param(param) tcsetattr(0, TCSANOW, param)
struct termios saved_tty_parameters;
#endif

char *abort_string[MAX_ABORTS], *fail_reason = (char *)0,
fail_buffer[50];
int n_aborts = 0, abort_next = 0, timeout_next = 0, echo_next = 0;
int clear_abort_next = 0;

char *report_string[MAX_REPORTS] ;
char report_buffer[256] ;
int n_reports = 0, report_next = 0, report_gathering = 0 ;
int clear_report_next = 0;

int say_next = 0, hup_next = 0;

void *dup_mem __P((void *b, size_t c));
void *copy_of __P((char *s));
char *grow __P((char *s, char **p, size_t len));
void usage __P((void));
void msgf __P((const char *fmt, ...));
void fatal __P((int code, const char *fmt, ...));
SIGTYPE sigalrm __P((int signo));
SIGTYPE sigint __P((int signo));
SIGTYPE sigterm __P((int signo));
SIGTYPE sighup __P((int signo));
void unalarm __P((void));
void init __P((void));

```

```

void set_tty_parameters __P((void));
void echo_stderr __P((int));
void break_sequence __P((void));
void terminate __P((int status));
void do_file __P((char *chat_file));
int get_string __P((register char *string));
int put_string __P((register char *s));
int write_char __P((int c));
int put_char __P((int c));
int get_char __P((void));
void chat_send __P((register char *s));
char *character __P((int c));
void chat_expect __P((register char *s));
char *clean __P((register char *s, int sending));
void break_sequence __P((void));
void terminate __P((int status));
void pack_array __P((char **array, int end));
char *expect_strtok __P((char *, char *));
int vfmtmsg __P((char *, int, const char *, va_list));/* vsprintf++ */

int main __P((int, char *[]));

void *dup_mem(b, c)
void *b;
size_t c;
{
    void *ans = malloc (c);
    if (!ans)
fatal(2, "memory error!");

    memcpy (ans, b, c);
    return ans;
}

void *copy_of (s)
char *s;
{
    return dup_mem (s, strlen (s) + 1);
}

/* grow a char buffer and keep a pointer offset */
char *grow(s, p, len)
char *s;
char **p;
size_t len;
{
    size_t l = *p - s; /* save p as distance into s */

    s = realloc(s, len);
    if (!s)
fatal(2, "memory error!");
    *p = s + l; /* restore p */
    return s;
}

```

```

/*
 * chat [ -v ] [ -E ] [ -T number ] [ -U number ] [ -t timeout ] [ -f chat-file ] \
 * [ -r report-file ] \
 * [...[[expect[-say[-expect...]] say expect[-say[-expect]] ...]]]
 *
 * Perform a UUCP-dialer-like chat script on stdin and stdout.
 */
int
main(argc, argv)
    int argc;
    char **argv;
{
    int option;
    char *arg;

    program_name = *argv;
    tzset();

    while ((option = OPTION(argc, argv)) != 0) {
switch (option) {
case 'e':
    ++echo;
    break;

case 'E':
    ++use_env;
    break;

case 'v':
    ++verbose;
    break;

case 'V':
    ++Verbose;
    break;

case 's':
    ++to_stderr;
    break;

case 'S':
    to_log = 0;
    break;

case 'f':
    if ((arg = OPTARG(argc, argv)) != NULL)
        chat_file = copy_of(arg);
    else
        usage();
    break;

case 't':
    if ((arg = OPTARG(argc, argv)) != NULL)
        timeout = atoi(arg);

```

```
    else
    usage();
    break;

case 'r':
    arg = OPTARG (argc, argv);
    if (arg) {
    if (report_fp != NULL)
        fclose (report_fp);
    report_file = copy_of (arg);
    report_fp = fopen (report_file, "a");
    if (report_fp != NULL) {
        if (verbose)
        fprintf (report_fp, "Opening \"%s\"...\n",
            report_file);
        report = 1;
    }
    }
    break;

case 'T':
    if ((arg = OPTARG(argc, argv)) != NULL)
    phone_num = copy_of(arg);
    else
    usage();
    break;

case 'U':
    if ((arg = OPTARG(argc, argv)) != NULL)
    phone_num2 = copy_of(arg);
    else
    usage();
    break;

default:
    usage();
    break;
}
}
/*
 * Default the report file to the stderr location
 */
    if (report_fp == NULL)
report_fp = stderr;

    if (to_log) {
#ifdef ultrix
    openlog("chat", LOG_PID);
#else
    openlog("chat", LOG_PID | LOG_NDELAY, LOG_LOCAL2);

    if (verbose)
        setlogmask(LOG_UPTO(LOG_INFO));
    else
```

```

    setlogmask(LOG_UPTO(LOG_WARNING));
#endif
}

init();

if (chat_file != NULL) {
    arg = ARG(argc, argv);
    if (arg != NULL)
        usage();
    else
        do_file (chat_file);
} else {
    while ((arg = ARG(argc, argv)) != NULL) {
        chat_expect(arg);

        if ((arg = ARG(argc, argv)) != NULL)
            chat_send(arg);
    }

    terminate(0);
    return 0;
}

/*
 * Process a chat script when read from a file.
 */

void do_file (chat_file)
char *chat_file;
{
    int linect, sendflg;
    char *sp, *arg, quote;
    char buf [STR_LEN];
    FILE *cfp;

    cfp = fopen (chat_file, "r");
    if (cfp == NULL)
        fatal(1, "%s -- open failed: %m", chat_file);

    linect = 0;
    sendflg = 0;

    while (fgets(buf, STR_LEN, cfp) != NULL) {
        sp = strchr (buf, '\n');
        if (sp)
            *sp = '\0';

        linect++;
        sp = buf;

        /* lines starting with '#' are comments. If a real '#'
           is to be expected, it should be quoted .... */
        if ( *sp == '#' )

```



```
        continue;

while (*sp != '\0') {
    if (*sp == ' ' || *sp == '\t') {
        ++sp;
        continue;
    }

    if (*sp == '"' || *sp == '\n') {
        quote = *sp++;
        arg = sp;
        while (*sp != quote) {
            if (*sp == '\0')
                fatal(1, "unterminated quote (line %d)", linect);

            if (*sp++ == '\\') {
                if (*sp != '\0')
                    ++sp;
            }
        }
        else {
            arg = sp;
            while (*sp != '\0' && *sp != ' ' && *sp != '\t')
                ++sp;
        }

        if (*sp != '\0')
            *sp++ = '\0';

        if (sendflg)
            chat_send (arg);
        else
            chat_expect (arg);
        sendflg = !sendflg;
    }
    }
    fclose (cfp);
}

/*
 *We got an error parsing the command line.
 */
void usage()
{
    fprintf(stderr, "\
Usage: %s [-e] [-E] [-v] [-V] [-t timeout] [-r report-file]\n\
        [-T phone-number] [-U phone-number2] {-f chat-file | chat-script}\n", program_name);
    exit(1);
}

char line[1024];

/*
```

```

    * Send a message to syslog and/or stderr.
    */
void msgf __V((const char *fmt, ...))
{
    va_list args;

#ifdef __STDC__
    va_start(args, fmt);
#else
    char *fmt;
    va_start(args);
    fmt = va_arg(args, char *);
#endif

    vfmtmsg(line, sizeof(line), fmt, args);
    if (to_log)
        syslog(LOG_INFO, "%s", line);
    if (to_stderr)
        fprintf(stderr, "%s\n", line);
}

/*
 * Print an error message and terminate.
 */

void fatal __V((int code, const char *fmt, ...))
{
    va_list args;

#ifdef __STDC__
    va_start(args, fmt);
#else
    int code;
    char *fmt;
    va_start(args);
    code = va_arg(args, int);
    fmt = va_arg(args, char *);
#endif

    vfmtmsg(line, sizeof(line), fmt, args);
    if (to_log)
        syslog(LOG_ERR, "%s", line);
    if (to_stderr)
        fprintf(stderr, "%s\n", line);
    terminate(code);
}

int alarmed = 0;

SIGTYPE sigalrm(signo)
int signo;
{
    int flags;

    alarm(1);

```

```
    alarmed = 1; /* Reset alarm to avoid race window */
    signal(SIGALRM, sigalrm); /* that can cause hanging in read() */

    if ((flags = fcntl(0, F_GETFL, 0)) == -1)
        fatal(2, "Can't get file mode flags on stdin: %m");

    if (fcntl(0, F_SETFL, flags | O_NONBLOCK) == -1)
        fatal(2, "Can't set file mode flags on stdin: %m");

    if (verbose)
        msgf("alarm");
}

void unalarm()
{
    int flags;

    if ((flags = fcntl(0, F_GETFL, 0)) == -1)
        fatal(2, "Can't get file mode flags on stdin: %m");

    if (fcntl(0, F_SETFL, flags & ~O_NONBLOCK) == -1)
        fatal(2, "Can't set file mode flags on stdin: %m");
}

SIGTYPE sigint(signo)
int signo;
{
    fatal(2, "SIGINT");
}

SIGTYPE sigterm(signo)
int signo;
{
    fatal(2, "SIGTERM");
}

SIGTYPE sighup(signo)
int signo;
{
    fatal(2, "SIGHUP");
}

void init()
{
    signal(SIGINT, sigint);
    signal(SIGTERM, sigterm);
    signal(SIGHUP, sighup);

    set_tty_parameters();
    signal(SIGALRM, sigalrm);
    alarm(0);
    alarmed = 0;
}
```

```

void set_tty_parameters()
{
#ifdef defined(get_term_param)
    term_parms t;

    if (get_term_param (&t) < 0)
        fatal(2, "Can't get terminal parameters: %m");

    saved_tty_parameters = t;
    have_tty_parameters = 1;

    t.c_iflag  |= IGNBRK | ISTRIP | IGNPAR;
    t.c_oflag   = 0;
    t.c_lflag   = 0;
    t.c_cc[VERASE] =
    t.c_cc[VKILL]  = 0;
    t.c_cc[VMIN]   = 1;
    t.c_cc[VTIME]  = 0;

    if (set_term_param (&t) < 0)
        fatal(2, "Can't set terminal parameters: %m");
#endif
}

void break_sequence()
{
#ifdef TERMIO
    tcsendbreak (0, 0);
#endif
}

void terminate(status)
int status;
{
    static int terminating = 0;

    if (terminating)
        exit(status);
    terminating = 1;
    echo_stderr(-1);
    /*
     * Allow the last of the report string to be gathered before we terminate.
     */
    if (report_gathering) {
        int c, rep_len;

        rep_len = strlen(report_buffer);
        while (rep_len + 1 <= sizeof(report_buffer)) {
            alarm(1);
            c = get_char();
            alarm(0);
            if (c < 0 || iscntrl(c))
                break;
            report_buffer[rep_len] = c;
            ++rep_len;
        }
    }
}

```

```

    }
    report_buffer[rep_len] = 0;
    fprintf(report_fp, "chat: %s\n", report_buffer);
    }
    if (report_file != (char *) 0 && report_fp != (FILE *) NULL) {
if (verbose)
    fprintf(report_fp, "Closing \"%s\".\n", report_file);
fclose (report_fp);
report_fp = (FILE *) NULL;
    }

#ifdef defined(get_term_param)
    if (have_tty_parameters) {
if (set_term_param (&saved_tty_parameters) < 0)
    fatal(2, "Can't restore terminal parameters: %m");
    }
#endif

    exit(status);
}

/*
 *'Clean up' this string.
 */
char *clean(s, sending)
register char *s;
int sending; /* set to 1 when sending (putting) this string. */
{
    char cur_chr;
    char *s1, *p, *phchar;
    int add_return = sending;
    size_t len = strlen(s) + 3; /* see len comments below */

#define isoctal(chr) (((chr) >= '0') && ((chr) <= '7'))
#define isalnumx(chr) (((chr) >= '0') && ((chr) <= '9')) \
    || (((chr) >= 'a') && ((chr) <= 'z')) \
    || (((chr) >= 'A') && ((chr) <= 'Z')) \
    || (chr) == '_'

    p = s1 = malloc(len);
    if (!p)
fatal(2, "memory error!");
    while (*s) {
cur_chr = *s++;
if (cur_chr == '^') {
    cur_chr = *s++;
    if (cur_chr == '\0') {
*p++ = '^';
break;
    }
    cur_chr &= 0x1F;
    if (cur_chr != 0) {
*p++ = cur_chr;
    }

```

```

        continue;
    }

    if (use_env && cur_chr == '$') { /* ARI */
        char c;

        phchar = s;
        while (isalnumx(*s))
            s++;
        c = *s; /* save */
        *s = '\0';
        phchar = getenv(phchar);
        *s = c; /* restore */
        if (phchar) {
            len += strlen(phchar);
            sl = grow(sl, &p, len);
            while (*phchar)
                *p++ = *phchar++;
        }
        continue;
    }

    if (cur_chr != '\\') {
        *p++ = cur_chr;
        continue;
    }

    cur_chr = *s++;
    if (cur_chr == '\0') {
        if (sending) {
            *p++ = '\\';
            *p++ = '\\'; /* +1 for len */
        }
        break;
    }

    switch (cur_chr) {
    case 'b':
        *p++ = '\b';
        break;

    case 'c':
        if (sending && *s == '\0')
            add_return = 0;
        else
            *p++ = cur_chr;
        break;

    case '\\':
    case 'K':
    case 'p':
    case 'd':
        if (sending)
            *p++ = '\\';
        *p++ = cur_chr;
    }

```

```
        break;

    case 'T':
        if (sending && phone_num) {
            len += strlen(phone_num);
            s1 = grow(s1, &p, len);
            for (phchar = phone_num; *phchar != '\0'; phchar++)
                *p++ = *phchar;
        }
        else {
            *p++ = "\\';
            *p++ = 'T';
        }
        break;

    case 'U':
        if (sending && phone_num2) {
            len += strlen(phone_num2);
            s1 = grow(s1, &p, len);
            for (phchar = phone_num2; *phchar != '\0'; phchar++)
                *p++ = *phchar;
        }
        else {
            *p++ = "\\';
            *p++ = 'U';
        }
        break;

    case 'q':
        quiet = 1;
        break;

    case 'r':
        *p++ = "\r";
        break;

    case 'n':
        *p++ = "\n";
        break;

    case 's':
        *p++ = ' ';
        break;

    case 't':
        *p++ = "\t";
        break;

    case 'N':
        if (sending) {
            *p++ = "\\';
            *p++ = '\0';
        }
        else
```

```

    *p++ = 'N';
    break;

case '$':/* ARI */
    if (use_env) {
        *p++ = cur_chr;
        break;
    }
    /* FALL THROUGH */

default:
    if (isoctal (cur_chr)) {
        cur_chr &= 0x07;
    }
    if (isoctal (*s)) {
        cur_chr <= 3;
        cur_chr |= *s++ - '0';
    }
    if (isoctal (*s)) {
        cur_chr <= 3;
        cur_chr |= *s++ - '0';
    }
    }

    if (cur_chr != 0 || sending) {
        if (sending && (cur_chr == '\\ || cur_chr == 0))
            *p++ = '\\';
        *p++ = cur_chr;
    }
    break;
}

    if (sending)
        *p++ = '\\';
    *p++ = cur_chr;
    break;
}

    if (add_return)
        *p++ = '\r';/* +2 for len */

    *p = '\0';/* +3 for len */
    return s1;
}

/*
 * A modified version of 'strtok'. This version skips \ sequences.
 */

char *expect_strtok (s, term)
    char *s, *term;
{
    static char *str = "";
    int escape_flag = 0;
    char *result;

```



```
/*
 * If a string was specified then do initial processing.
 */
    if (s)
str = s;

/*
 * If this is the escape flag then reset it and ignore the character.
 */
    if (*str)
result = str;
    else
result = (char *) 0;

    while (*str) {
if (escape_flag) {
    escape_flag = 0;
    ++str;
    continue;
}

if (*str == '\\') {
    ++str;
    escape_flag = 1;
    continue;
}

/*
 * If this is not in the termination string, continue.
 */
if (strchr (term, *str) == (char *) 0) {
    ++str;
    continue;
}

/*
 * This is the terminator. Mark the end of the string and stop.
 */
*str++ = '\0';
break;
    }
    return (result);
}

/*
 * Process the expect string
 */

void chat_expect (s)
char *s;
{
    char *expect;
    char *reply;
```

```
    if (strcmp(s, "HANGUP") == 0) {
        ++hup_next;
        return;
    }

    if (strcmp(s, "ABORT") == 0) {
        ++abort_next;
        return;
    }

    if (strcmp(s, "CLR_ABORT") == 0) {
        ++clear_abort_next;
        return;
    }

    if (strcmp(s, "REPORT") == 0) {
        ++report_next;
        return;
    }

    if (strcmp(s, "CLR_REPORT") == 0) {
        ++clear_report_next;
        return;
    }

    if (strcmp(s, "TIMEOUT") == 0) {
        ++timeout_next;
        return;
    }

    if (strcmp(s, "ECHO") == 0) {
        ++echo_next;
        return;
    }

    if (strcmp(s, "SAY") == 0) {
        ++say_next;
        return;
    }

    /*
     * Fetch the expect and reply string.
     */
    for (;;) {
        expect = expect_strtok (s, "-");
        s = (char *) 0;

        if (expect == (char *) 0)
            return;

        reply = expect_strtok (s, "-");

        /*
         * Handle the expect string. If successful then exit.
         */
    }
```

```
if (get_string (expect))
    return;

/*
 * If there is a sub-reply string then send it. Otherwise any condition
 * is terminal.
 */
if (reply == (char *) 0 || exit_code != 3)
    break;

chat_send (reply);
}

/*
 * The expectation did not occur. This is terminal.
 */
if (fail_reason)
    msgf("Failed (%s)", fail_reason);
else
    msgf("Failed");
terminate(exit_code);
}

/*
 * Translate the input character to the appropriate string for printing
 * the data.
 */

char *character(c)
int c;
{
    static char string[10];
    char *meta;

    meta = (c & 0x80) ? "M-" : "";
    c &= 0x7F;

    if (c < 32)
        sprintf(string, "%s^%c", meta, (int)c + '@');
    else if (c == 127)
        sprintf(string, "%s^?", meta);
    else
        sprintf(string, "%s%c", meta, c);

    return (string);
}

/*
 * process the reply string
 */
void chat_send (s)
register char *s;
{
    char file_data[STR_LEN];
```

```

    if (say_next) {
say_next = 0;
s = clean(s, 1);
write(2, s, strlen(s));
    free(s);
return;
    }

    if (hup_next) {
        hup_next = 0;
if (strcmp(s, "OFF") == 0)
    signal(SIGHUP, SIG_IGN);
        else
            signal(SIGHUP, sighup);
        return;
    }

    if (echo_next) {
echo_next = 0;
echo = (strcmp(s, "ON") == 0);
return;
    }

    if (abort_next) {
char *s1;

abort_next = 0;

if (n_aborts >= MAX_ABORTS)
    fatal(2, "Too many ABORT strings");

s1 = clean(s, 0);

if (strlen(s1) > strlen(s)
    || strlen(s1) + 1 > sizeof(fail_buffer))
    fatal(1, "Illegal or too-long ABORT string ('%v')", s);

abort_string[n_aborts++] = s1;

if (verbose)
    msgf("abort on (%v)", s);
return;
    }

    if (clear_abort_next) {
char *s1;
int i;
    int old_max;
int pack = 0;

clear_abort_next = 0;

s1 = clean(s, 0);

```

```
if (strlen(s1) > strlen(s)
    || strlen(s1) + 1 > sizeof(fail_buffer))
    fatal(1, "Illegal or too-long CLR_ABORT string ('%v'", s);

    old_max = n_aborts;
    for (i=0; i < n_aborts; i++) {
        if (strcmp(s1, abort_string[i]) == 0) {
            free(abort_string[i]);
            abort_string[i] = NULL;
            pack++;
            n_aborts--;
            if (verbose)
                msgf("clear abort on (%v)", s);
        }
    }
    free(s1);
    if (pack)
        pack_array(abort_string, old_max);
    return;
}

if (report_next) {
    char *s1;

    report_next = 0;
    if (n_reports >= MAX_REPORTS)
        fatal(2, "Too many REPORT strings");

    s1 = clean(s, 0);
    if (strlen(s1) > strlen(s)
        || strlen(s1) + 1 > sizeof(fail_buffer))
        fatal(1, "Illegal or too-long REPORT string ('%v'", s);

    report_string[n_reports++] = s1;

    if (verbose)
        msgf("report (%v)", s);
    return;
}

if (clear_report_next) {
    char *s1;
    int i;
    int old_max;
    int pack = 0;

    clear_report_next = 0;

    s1 = clean(s, 0);

    if (strlen(s1) > strlen(s)
        || strlen(s1) + 1 > sizeof(fail_buffer))
        fatal(1, "Illegal or too-long REPORT string ('%v'", s);
```

```

    old_max = n_reports;
    for (i=0; i < n_reports; i++) {
        if ( strcmp(s1,report_string[i]) == 0 ) {
            free(report_string[i]);
            report_string[i] = NULL;
            pack++;
            n_reports--;
            if (verbose)
                msgf("clear report (%v)", s);
        }
    }

    free(s1);
    if (pack)
        pack_array(report_string,old_max);

return;
}

    if (timeout_next) {
        timeout_next = 0;
        s = clean(s, 0);
        timeout = atoi(s);

    if (timeout <= 0)
        timeout = DEFAULT_CHAT_TIMEOUT;

    if (verbose)
        msgf("timeout set to %d seconds", timeout);

return;
}

/*
 * The syntax @filename means read the string to send from the
 * file 'filename'.
 */
    if (s[0] == '@') {
/* skip the @ and any following white-space */
        char *fn = s;
        while (*++fn == ' ' || *fn == '\t')
            ;

        if (*fn != 0) {
            FILE *f;
            int n = 0;

            /* open the file and read until STR_LEN-1 bytes or end-of-file */
            f = fopen(fn, "r");
            if (f == NULL)
                fatal(1, "%s -- open failed: %m", fn);
            while (n < STR_LEN - 1) {
                int nr = fread(&file_data[n], 1, STR_LEN - 1 - n, f);
                if (nr < 0)
                    fatal(1, "%s -- read error", fn);
                if (nr == 0)

```

```
        break;
    n += nr;
    }
    fclose(f);

    /* use the string we got as the string to send,
       but trim off the final newline if any. */
    if (n > 0 && file_data[n-1] == '\n')
--n;
    file_data[n] = 0;
    s = file_data;
}
}

    if (strcmp(s, "EOT") == 0)
s = "^D\\c";
    else if (strcmp(s, "BREAK") == 0)
s = "\\K\\c";

    if (!put_string(s))
fatal(1, "Failed");
}

int get_char()
{
    int status;
    char c;

    status = read(0, &c, 1);

    switch (status) {
        case 1:
return ((int)c & 0x7F);

        default:
msgf("warning: read() on stdin returned %d", status);

        case -1:
if ((status = fcntl(0, F_GETFL, 0)) == -1)
    fatal(2, "Can't get file mode flags on stdin: %m");

if (fcntl(0, F_SETFL, status & ~O_NONBLOCK) == -1)
    fatal(2, "Can't set file mode flags on stdin: %m");

return (-1);
    }
}

int put_char(c)
int c;
{
    int status;
    char ch = c;
```

```

        usleep(10000);/* inter-character typing delay (?) */

        status = write(1, &ch, 1);

        switch (status) {
        case 1:
        return (0);

        default:
        msgf("warning: write() on stdout returned %d", status);

        case -1:
        if ((status = fcntl(0, F_GETFL, 0)) == -1)
            fatal(2, "Can't get file mode flags on stdin, %m");

        if (fcntl(0, F_SETFL, status & ~O_NONBLOCK) == -1)
            fatal(2, "Can't set file mode flags on stdin: %m");

        return (-1);
        }
    }

    int write_char (c)
    int c;
    {
        if (alarmed || put_char(c) < 0) {
        alarm(0);
        alarmed = 0;

        if (verbose) {
            if (errno == EINTR || errno == EWOULDBLOCK)
            msgf(" -- write timed out");
            else
            msgf(" -- write failed: %m");
        }
        return (0);
        }
        return (1);
    }

    int put_string (s)
    register char *s;
    {
        quiet = 0;
        s = clean(s, 1);

        if (verbose) {
        if (quiet)
            msgf("send (????\?)");
        else
            msgf("send (%v)", s);
        }

        alarm(timeout); alarmed = 0;
    }

```



```
    while (*s) {
register char c = *s++;

    if (c != "\\") {
        if (!write_char (c))
            return 0;
        continue;
    }

    c = *s++;
    switch (c) {
    case 'd':
        sleep(1);
        break;

    case 'K':
        break_sequence();
        break;

    case 'p':
        usleep(10000); /* 1/100th of a second (arg is microseconds) */
        break;

    default:
        if (!write_char (c))
            return 0;
        break;
    }
}

alarm(0);
alarmed = 0;
return (1);
}

/*
 *Echo a character to stderr.
 *When called with -1, a '\n' character is generated when
 *the cursor is not at the beginning of a line.
 */
void echo_stderr(n)
int n;
{
    static int need_lf;
    char *s;

    switch (n) {
    case '\r': /* ignore '\r' */
        break;
    case -1:
        if (need_lf == 0)
            break;
        /* fall through */
    case '\n':
```

```

write(2, "\n", 1);
need_lf = 0;
break;
    default:
s = character(n);
write(2, s, strlen(s));
need_lf = 1;
break;
    }
}

/*
*'Wait for' this string to appear on this file descriptor.
*/
int get_string(string)
register char *string;
{
    char temp[STR_LEN];
    int c, printed = 0, len, minlen;
    register char *s = temp, *end = s + STR_LEN;
    char *logged = temp;

    fail_reason = (char *)0;
    string = clean(string, 0);
    len = strlen(string);
    minlen = (len > sizeof(fail_buffer)? len: sizeof(fail_buffer)) - 1;

    if (verbose)
msgf("expect (%v)", string);

    if (len > STR_LEN) {
msgf("expect string is too long");
exit_code = 1;
return 0;
    }

    if (len == 0) {
if (verbose)
    msgf("got it");
return (1);
    }

    alarm(timeout);
    alarmed = 0;

    while ( ! alarmed && (c = get_char()) >= 0) {
int n, abort_len, report_len;

if (echo)
    echo_stderr(c);
if (verbose && c == '\n') {
    if (s == logged)
msgf("");/* blank line */
    else
msgf("%0.*v", s - logged, logged);

```

```

        logged = s + 1;
    }

    *s++ = c;

    if (verbose && s >= logged + 80) {
        msgf("%0.*v", s - logged, logged);
        logged = s;
    }

    if (Verbose) {
        if (c == '\n')
            fputc( '\n', stderr );
        else if (c != '\r')
            fprintf( stderr, "%s", character(c) );
    }

    if (!report_gathering) {
        for (n = 0; n < n_reports; ++n) {
            if ((report_string[n] != (char*) NULL) &&
                s - temp >= (report_len = strlen(report_string[n])) &&
                strncmp(s - report_len, report_string[n], report_len) == 0) {
                time_t time_now = time ((time_t*) NULL);
                struct tm* tm_now = localtime (&time_now);

                strftime (report_buffer, 20, "%b %d %H:%M:%S ", tm_now);
                strcat (report_buffer, report_string[n]);

                report_string[n] = (char *) NULL;
                report_gathering = 1;
                break;
            }
        }
    }
    else {
        if (!iscntrl (c)) {
            int rep_len = strlen (report_buffer);
            report_buffer[rep_len] = c;
            report_buffer[rep_len + 1] = '\0';
        }
        else {
            report_gathering = 0;
            fprintf (report_fp, "chat: %s\n", report_buffer);
        }
    }

    if (s - temp >= len &&
        c == string[len - 1] &&
        strncmp(s - len, string, len) == 0) {
        if (verbose) {
            if (s > logged)
                msgf("%0.*v", s - logged, logged);
            msgf(" -- got it\n");
        }
    }

```

```

    alarm(0);
    alarmed = 0;
    return (1);
}

for (n = 0; n < n_aborts; ++n) {
    if (s - temp >= (abort_len = strlen(abort_string[n])) &&
        strncmp(s - abort_len, abort_string[n], abort_len) == 0) {
        if (verbose) {
            if (s > logged)
                msgf("%0.*v", s - logged, logged);
            msgf(" -- failed");
        }

        alarm(0);
        alarmed = 0;
        exit_code = n + 4;
        strcpy(fail_reason = fail_buffer, abort_string[n]);
        return (0);
    }

    if (s >= end) {
        if (logged < s - minlen) {
            if (verbose)
                msgf("%0.*v", s - logged, logged);
            logged = s;
        }
        s -= minlen;
        memmove(temp, s, minlen);
        logged = temp + (logged - s);
        s = temp + minlen;
    }

    if (alarmed && verbose)
        msgf("warning: alarm synchronization problem");
}

alarm(0);

if (verbose && printed) {
    if (alarmed)
        msgf(" -- read timed out");
    else
        msgf(" -- read failed: %m");
}

exit_code = 3;
alarmed = 0;
return (0);
}

/*
 * Gross kludge to handle Solaris versions >= 2.6 having usleep.

```

```

*/
#ifdef SOL2
#include <sys/param.h>
#if MAXUID > 65536 /* then this is Solaris 2.6 or later */
#undef NO_USLEEP
#endif
#endif /* SOL2 */

#ifdef NO_USLEEP
#include <sys/types.h>
#include <sys/time.h>

/*
usleep -- support routine for 4.2BSD system call emulations
last edit: 29-Oct-1984   D A Gwyn
*/

extern int select();

int
usleep( usec ) /* returns 0 if ok, else -1 */
{
    long usec; /* delay in microseconds */
    {
        static struct { /* `timeval' */
            long tv_sec; /* seconds */
            long tv_usec; /* microsecs */
        } delay; /* _select() timeout */

        delay.tv_sec = usec / 1000000L;
        delay.tv_usec = usec % 1000000L;

        return select(0, (long *)0, (long *)0, (long *)0, &delay);
    }
}
#endif

void
pack_array (array, end)
char **array; /* The address of the array of string pointers */
int end; /* The index of the next free entry before CLR_ */
{
    int i, j;

    for (i = 0; i < end; i++) {
        if (array[i] == NULL) {
            for (j = i+1; j < end; ++j)
                if (array[j] != NULL)
                    array[i++] = array[j];
            for (; i < end; ++i)
                array[i] = NULL;
            break;
        }
    }
}

```

```

/*
 * vfntmsg - format a message into a buffer. Like vsprintf except we
 * also specify the length of the output buffer, and we handle the
 * %m (error message) format.
 * Doesn't do floating-point formats.
 * Returns the number of chars put into buf.
 */
#define OUTCHAR(c)(buflen > 0? (--buflen, *buf++ = (c)): 0)

int
vfntmsg(buf, buflen, fmt, args)
    char *buf;
    int buflen;
    const char *fmt;
    va_list args;
{
    int c, i, n;
    int width, prec, fillch;
    int base, len, neg, quoted;
    unsigned long val = 0;
    char *str, *buf0;
    const char *f;
    unsigned char *p;
    char num[32];
    static char hexchars[] = "0123456789abcdef";

    buf0 = buf;
    --buflen;
    while (buflen > 0) {
        for (f = fmt; *f != '%' && *f != 0; ++f)
            ;
        if (f > fmt) {
            len = f - fmt;
            if (len > buflen)
                len = buflen;
            memcpy(buf, f, len);
            buf += len;
            buflen -= len;
            f = f;
        }
        if (*f == 0)
            break;
        c = *++f;
        width = prec = 0;
        fillch = ' ';
        if (c == '0') {
            fillch = '0';
            c = *++f;
        }
        if (c == '*') {
            width = va_arg(args, int);
            c = *++f;
        }
        else {
            while (isdigit(c)) {
                width = width * 10 + c - '0';
            }
        }
    }

```

```
c = *++fmt;
    }
}
if (c == '.') {
    c = *++fmt;
    if (c == '*') {
prec = va_arg(args, int);
c = *++fmt;
    } else {
while (isdigit(c)) {
    prec = prec * 10 + c - '0';
    c = *++fmt;
}
    }
}
str = 0;
base = 0;
neg = 0;
++fmt;
switch (c) {
case 'd':
    i = va_arg(args, int);
    if (i < 0) {
neg = 1;
val = -i;
    } else
val = i;
    base = 10;
    break;
case 'o':
    val = va_arg(args, unsigned int);
    base = 8;
    break;
case 'x':
    val = va_arg(args, unsigned int);
    base = 16;
    break;
case 'p':
    val = (unsigned long) va_arg(args, void *);
    base = 16;
    neg = 2;
    break;
case 's':
    str = va_arg(args, char *);
    break;
case 'c':
    num[0] = va_arg(args, int);
    num[1] = 0;
    str = num;
    break;
case 'm':
    str = strerror(errno);
    break;
case 'v':/* "visible" string */
```

```

case 'q':/* quoted string */
    quoted = c == 'q';
    p = va_arg(args, unsigned char *);
    if (fillch == '0' && prec > 0) {
n = prec;
    } else {
n = strlen((char *)p);
    if (prec > 0 && prec < n)
        n = prec;
    }
    while (n > 0 && buflen > 0) {
c = *p++;
--n;
    if (!quoted && c >= 0x80) {
        OUTCHAR('M');
        OUTCHAR('-');
        c -= 0x80;
    }
    if (quoted && (c == '"' || c == '\\'))
        OUTCHAR('\\');
    if (c < 0x20 || (0x7f <= c && c < 0xa0)) {
        if (quoted) {
            OUTCHAR('\\');
            switch (c) {
            case '\t':OUTCHAR('t');break;
            case '\n':OUTCHAR('n');break;
            case '\b':OUTCHAR('b');break;
            case '\f':OUTCHAR('f');break;
            default:
                OUTCHAR('x');
                OUTCHAR(hexchars[c >> 4]);
                OUTCHAR(hexchars[c & 0xf]);
            }
        } else {
            if (c == '\t')
                OUTCHAR(c);
            else {
                OUTCHAR('^');
                OUTCHAR(c ^ 0x40);
            }
        }
    } else
        OUTCHAR(c);
    continue;
default:
    *buf++ = '%';
    if (c != '%')
--fmt;/* so %z outputs %z etc. */
    --buflen;
    continue;
}
if (base != 0) {
    str = num + sizeof(num);
    *--str = 0;
}

```



```

    while (str > num + neg) {
        *--str = hexchars[val % base];
        val = val / base;
        if (--prec <= 0 && val == 0)
            break;
    }
    switch (neg) {
        case 1:
            *--str = '-';
            break;
        case 2:
            *--str = 'x';
            *--str = '0';
            break;
    }
    len = num + sizeof(num) - 1 - str;
} else {
    len = strlen(str);
    if (prec > 0 && len > prec)
        len = prec;
}
if (width > 0) {
    if (width > buflen)
        width = buflen;
    if ((n = width - len) > 0) {
        buflen -= n;
        for (; n > 0; --n)
            *buf++ = fillch;
    }
}
if (len > buflen)
    len = buflen;
memcpy(buf, str, len);
buf += len;
buflen -= len;
}
*buf = 0;
return buf - buf0;
}

```

Copyright (c) The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.

Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LIBRARY GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies

of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is

numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a

medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library),

you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License

which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source

file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the library's name and an idea of what it does.

Copyright (C) year name of author

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Library General Public
License as published by the Free Software Foundation; either
version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Library General Public License for more details.

You should have received a copy of the GNU Library General Public
License along with this library; if not, write to the
Free Software Foundation, Inc., 51 Franklin St, Fifth Floor,
Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in
the library 'Frob' (a library for tweaking knobs) written
by James Random Hacker.

signature of Ty Coon, 1 April 1990

Ty Coon, President of Vice

That's all there is to it!

This is a placeholder for the Public Domain License

```

/*
 * passprompt.c - pppd plugin to invoke an external PAP password prompter
 *
 * Copyright 1999 Paul Mackerras, Alan Curry.
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version
 * 2 of the License, or (at your option) any later version.
 */
#include <errno.h>
#include <unistd.h>
#include <sys/wait.h>
#include <syslog.h>
#include "pppd.h"

char pppd_version[] = VERSION;

static char promptprog[PATH_MAX+1];

static option_t options[] = {
    { "promptprog", o_string, promptprog,
      "External PAP password prompting program",
      OPT_STATIC, NULL, PATH_MAX },
    { NULL }
};

static int promptpass(char *user, char *passwd)
{
    int p[2];
    pid_t kid;
    int readgood, wstat;
    ssize_t red;

    if (promptprog[0] == 0 || access(promptprog, X_OK) < 0)
        return -1; /* sorry, can't help */

    if (!passwd)
        return 1;

    if (pipe(p)) {
        warn("Can't make a pipe for %s", promptprog);
        return 0;
    }
    if ((kid = fork()) == (pid_t) -1) {
        warn("Can't fork to run %s", promptprog);
    }

```



```
close(p[0]);
close(p[1]);
return 0;
}
if (!kid) {
/* we are the child, exec the program */
char *argv[5], fdstr[32];
sys_close();
closelog();
close(p[0]);
seteuid(getuid());
setegid(getgid());
argv[0] = promptprog;
argv[1] = user;
argv[2] = remote_name;
sprintf(fdstr, "%d", p[1]);
argv[3] = fdstr;
argv[4] = 0;
execv(*argv, argv);
_exit(127);
}

/* we are the parent, read the password from the pipe */
close(p[1]);
readgood = 0;
do {
red = read(p[0], passwd + readgood, MAXSECRETLEN-1 - readgood);
if (red == 0)
break;
if (red < 0) {
if (errno == EINTR)
continue;
error("Can't read secret from %s: %m", promptprog);
readgood = -1;
break;
}
readgood += red;
} while (readgood < MAXSECRETLEN - 1);
close(p[0]);

/* now wait for child to exit */
while (waitpid(kid, &wstat, 0) < 0) {
if (errno != EINTR) {
warn("error waiting for %s: %m", promptprog);
break;
}
}

if (readgood < 0)
return 0;
passwd[readgood] = 0;
if (!WIFEXITED(wstat))
warn("%s terminated abnormally", promptprog);
if (WEXITSTATUS(wstat))
```

```
warn("%s exited with code %d", promptprog, WEXITSTATUS(status));
```

```
    return 1;
}
```

```
void plugin_init(void)
{
    add_options(options);
    pap_passwd_hook = promptpass;
}
/*
```

Unix SMB/CIFS implementation.

trivial database library

Copyright (C) Andrew Tridgell 1999-2004
 Copyright (C) Paul 'Rusty' Russell 2000
 Copyright (C) Jeremy Allison 2000-2003

**** NOTE! The following LGPL license applies to the tdb
 ** library. This does NOT imply that all of Samba is released
 ** under the LGPL**

This library is free software; you can redistribute it and/or
 modify it under the terms of the GNU Lesser General Public
 License as published by the Free Software Foundation; either
 version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
 License along with this library; if not, write to the Free Software
 Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

```
*/
```

```
/* NOTE: If you use tdb's under valgrind, and in particular if you run
 * tdbtorture, you may get spurious "uninitialized value" warnings. I
 * think this is because valgrind doesn't understand that the mmap'd
 * area may be written to by other processes. Memory can, from the
 * point of view of the grinded process, spontaneously become
 * initialized.
 *
 * I can think of a few solutions. [mbp 20030311]
 *
 * 1 - Write suppressions for Valgrind so that it doesn't complain
 * about this. Probably the most reasonable but people need to
 * remember to use them.
 *
 * 2 - Use IO not mmap when running under valgrind. Not so nice.
 *
 * 3 - Use the special valgrind macros to mark memory as valid at the
```

```

* right time. Probably too hard -- the process just doesn't know.
*/

#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <signal.h>
#include "tdb.h"
#include "spinlock.h"

#define TDB_MAGIC_FOOD "TDB file\n"
#define TDB_VERSION (0x26011967 + 6)
#define TDB_MAGIC (0x26011999U)
#define TDB_FREE_MAGIC (~TDB_MAGIC)
#define TDB_DEAD_MAGIC (0xFEE1DEAD)
#define TDB_ALIGNMENT 4
#define MIN_REC_SIZE (2*sizeof(struct list_struct) + TDB_ALIGNMENT)
#define DEFAULT_HASH_SIZE 131
#define TDB_PAGE_SIZE 0x2000
#define FREELIST_TOP (sizeof(struct tdb_header))
#define TDB_ALIGN(x,a) (((x) + (a)-1) & ~(a)-1)
#define TDB_BYTEREV(x) (((((x)&0xff)<<24)|((x)&0xFF00)<<8)|(((x)>>8)&0xFF00)|((x)>>24))
#define TDB_DEAD(r) ((r)->magic == TDB_DEAD_MAGIC)
#define TDB_BAD_MAGIC(r) ((r)->magic != TDB_MAGIC && !TDB_DEAD(r))
#define TDB_HASH_TOP(hash) (FREELIST_TOP + (BUCKET(hash)+1)*sizeof(tdb_off))
#define TDB_DATA_START(hash_size) (TDB_HASH_TOP(hash_size-1) + TDB_SPINLOCK_SIZE(hash_
size))

/* NB assumes there is a local variable called "tdb" that is the
 * current context, also takes doubly-parenthesized print-style
 * argument. */
#define TDB_LOG(x) (tdb->log_fn?((tdb->log_fn x),0) : 0)

/* lock offsets */
#define GLOBAL_LOCK 0
#define ACTIVE_LOCK 4

#ifndef MAP_FILE
#define MAP_FILE 0
#endif

#ifndef MAP_FAILED
#define MAP_FAILED ((void *)-1)
#endif

/* free memory if the pointer is valid and zero the pointer */
#ifndef SAFE_FREE

```

```

#define SAFE_FREE(x) do { if ((x) != NULL) {free((x)); (x)=NULL;} } while(0)
#endif

#define BUCKET(hash) ((hash) % tdb->header.hash_size)
TDB_DATA tdb_null;

/* all contexts, to ensure no double-opens (fcntl locks don't nest!) */
static TDB_CONTEXT *tdbs = NULL;

static int tdb_munmap(TDB_CONTEXT *tdb)
{
    if (tdb->flags & TDB_INTERNAL)
        return 0;

#ifdef HAVE_MMAP
    if (tdb->map_ptr) {
        int ret = munmap(tdb->map_ptr, tdb->map_size);
        if (ret != 0)
            return ret;
    }
#endif
    tdb->map_ptr = NULL;
    return 0;
}

static void tdb_mmap(TDB_CONTEXT *tdb)
{
    if (tdb->flags & TDB_INTERNAL)
        return;

#ifdef HAVE_MMAP
    if (!(tdb->flags & TDB_NOMMAP)) {
        tdb->map_ptr = mmap(NULL, tdb->map_size,
            PROT_READ|(tdb->read_only? 0:PROT_WRITE),
            MAP_SHARED|MAP_FILE, tdb->fd, 0);
    }

    /*
     * NB. When mmap fails it returns MAP_FAILED *NOT* NULL !!!!
     */

    if (tdb->map_ptr == MAP_FAILED) {
        tdb->map_ptr = NULL;
        TDB_LOG((tdb, 2, "tdb_mmap failed for size %d (%s)\n",
            tdb->map_size, strerror(errno)));
    }
    else {
        tdb->map_ptr = NULL;
    }
#else
    tdb->map_ptr = NULL;
#endif
}

/* Endian conversion: we only ever deal with 4 byte quantities */
static void *convert(void *buf, u32 size)

```

```

{
u32 i, *p = buf;
for (i = 0; i < size / 4; i++)
p[i] = TDB_BYTEREV(p[i]);
return buf;
}
#define DOCONV() (tdb->flags & TDB_CONVERT)
#define CONVERT(x) (DOCONV() ? convert(&x, sizeof(x)) : &x)

/* the body of the database is made of one list_struct for the free space
   plus a separate data list for each hash value */
struct list_struct {
tdb_off next; /* offset of the next record in the list */
tdb_len rec_len; /* total byte length of record */
tdb_len key_len; /* byte length of key */
tdb_len data_len; /* byte length of data */
u32 full_hash; /* the full 32 bit hash of the key */
u32 magic; /* try to catch errors */
/* the following union is implied:
union {
char record[rec_len];
struct {
char key[key_len];
char data[data_len];
}
}
u32 totalsize; (tailer)
}
*/
};

/*****
Allow a caller to set a "alarm" flag that tdb can check to abort
a blocking lock on SIGALRM.
*****/

static sig_atomic_t *palarm_fired;

void tdb_set_lock_alarm(sig_atomic_t *palarm)
{
palarm_fired = palarm;
}

/* a byte range locking function - return 0 on success
   this functions locks/unlocks 1 byte at the specified offset.

   On error, errno is also set so that errors are passed back properly
   through tdb_open(). */
static int tdb_brlock(TDB_CONTEXT *tdb, tdb_off offset,
int rw_type, int lck_type, int probe)
{
struct flock fl;
int ret;

if (tdb->flags & TDB_NOLOCK)

```

```

return 0;
if ((rw_type == F_WRLCK) && (tdb->read_only)) {
    errno = EACCES;
    return -1;
}

fl.l_type = rw_type;
fl.l_whence = SEEK_SET;
fl.l_start = offset;
fl.l_len = 1;
fl.l_pid = 0;

do {
    ret = fcntl(tdb->fd, lck_type, &fl);
    if (ret == -1 && errno == EINTR && palarm_fired && *palarm_fired)
        break;
} while (ret == -1 && errno == EINTR);

if (ret == -1) {
    if (!probe && lck_type != F_SETLK) {
        /* Ensure error code is set for log fun to examine. */
        if (errno == EINTR && palarm_fired && *palarm_fired)
            tdb->ecode = TDB_ERR_LOCK_TIMEOUT;
        else
            tdb->ecode = TDB_ERR_LOCK;
        TDB_LOG((tdb, 5, "tdb_brlck failed (fd=%d) at offset %d rw_type=%d lck_type=%d\n",
            tdb->fd, offset, rw_type, lck_type));
    }
    /* Was it an alarm timeout ? */
    if (errno == EINTR && palarm_fired && *palarm_fired) {
        TDB_LOG((tdb, 5, "tdb_brlck timed out (fd=%d) at offset %d rw_type=%d lck_type=%d\n",
            tdb->fd, offset, rw_type, lck_type));
        return TDB_ERRCODE(TDB_ERR_LOCK_TIMEOUT, -1);
    }
    /* Otherwise - generic lock error. errno set by fcntl.
     * EAGAIN is an expected return from non-blocking
     * locks. */
    if (errno != EAGAIN) {
        TDB_LOG((tdb, 5, "tdb_brlck failed (fd=%d) at offset %d rw_type=%d lck_type=%d: %s\n",
            tdb->fd, offset, rw_type, lck_type,
            strerror(errno)));
    }
    return TDB_ERRCODE(TDB_ERR_LOCK, -1);
}
return 0;
}

/* lock a list in the database. list -1 is the alloc list */
static int tdb_lock(TDB_CONTEXT *tdb, int list, int ltype)
{
    {
        if (list < -1 || list >= (int)tdb->header.hash_size) {
            TDB_LOG((tdb, 0, "tdb_lock: invalid list %d for ltype=%d\n",
                list, ltype));
            return -1;
        }
    }
}

```

```

if (tdb->flags & TDB_NOLOCK)
return 0;

/* Since fcntl locks don't nest, we do a lock for the first one,
   and simply bump the count for future ones */
if (tdb->locked[list+1].count == 0) {
if (!tdb->read_only && tdb->header.rwlocks) {
if (tdb_spinlock(tdb, list, ltype)) {
TDB_LOG((tdb, 0, "tdb_lock spinlock failed on list %d ltype=%d\n",
list, ltype));
return -1;
}
} else if (tdb_brlock(tdb, FREELIST_TOP+4*list, ltype, F_SETLKW, 0)) {
TDB_LOG((tdb, 0, "tdb_lock failed on list %d ltype=%d (%s)\n",
list, ltype, strerror(errno)));
return -1;
}
tdb->locked[list+1].ltype = ltype;
}
tdb->locked[list+1].count++;
return 0;
}

/* unlock the database: returns void because it's too late for errors. */
/* changed to return int it may be interesting to know there
   has been an error --simo */
static int tdb_unlock(TDB_CONTEXT *tdb, int list, int ltype)
{
int ret = -1;

if (tdb->flags & TDB_NOLOCK)
return 0;

/* Sanity checks */
if (list < -1 || list >= (int)tdb->header.hash_size) {
TDB_LOG((tdb, 0, "tdb_unlock: list %d invalid (%d)\n", list, tdb->header.hash_size));
return ret;
}

if (tdb->locked[list+1].count==0) {
TDB_LOG((tdb, 0, "tdb_unlock: count is 0\n"));
return ret;
}

if (tdb->locked[list+1].count == 1) {
/* Down to last nested lock: unlock underneath */
if (!tdb->read_only && tdb->header.rwlocks) {
ret = tdb_spinunlock(tdb, list, ltype);
} else {
ret = tdb_brlock(tdb, FREELIST_TOP+4*list, F_UNLCK, F_SETLKW, 0);
}
} else {
ret = 0;
}
}

```

```

tdb->locked[list+1].count--;

if (ret)
TDB_LOG((tdb, 0,"tdb_unlock: An error occurred unlocking!\n"));
return ret;
}

/* check for an out of bounds access - if it is out of bounds then
   see if the database has been expanded by someone else and expand
   if necessary
   note that "len" is the minimum length needed for the db
*/
static int tdb_oob(TDB_CONTEXT *tdb, tdb_off len, int probe)
{
struct stat st;
if (len <= tdb->map_size)
return 0;
if (tdb->flags & TDB_INTERNAL) {
if (!probe) {
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_IO;
TDB_LOG((tdb, 0,"tdb_oob len %d beyond internal malloc size %d\n",
(int)len, (int)tdb->map_size));
}
return TDB_ERRCODE(TDB_ERR_IO, -1);
}

if (fstat(tdb->fd, &st) == -1)
return TDB_ERRCODE(TDB_ERR_IO, -1);

if (st.st_size < (size_t)len) {
if (!probe) {
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_IO;
TDB_LOG((tdb, 0,"tdb_oob len %d beyond eof at %d\n",
(int)len, (int)st.st_size));
}
return TDB_ERRCODE(TDB_ERR_IO, -1);
}

/* Unmap, update size, remap */
if (tdb_munmap(tdb) == -1)
return TDB_ERRCODE(TDB_ERR_IO, -1);
tdb->map_size = st.st_size;
tdb_mmap(tdb);
return 0;
}

/* write a lump of data at a specified offset */
static int tdb_write(TDB_CONTEXT *tdb, tdb_off off, void *buf, tdb_len len)
{
if (tdb_oob(tdb, off + len, 0) != 0)
return -1;

if (tdb->map_ptr)

```



```

memcpy(off + (char *)tdb->map_ptr, buf, len);
#ifdef HAVE_PWRITE
else if (pwrite(tdb->fd, buf, len, off) != (ssize_t)len) {
#else
else if (lseek(tdb->fd, off, SEEK_SET) != off
|| write(tdb->fd, buf, len) != (ssize_t)len) {
#endif
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_IO;
TDB_LOG((tdb, 0, "tdb_write failed at %d len=%d (%s)\n",
off, len, strerror(errno)));
return TDB_ERRCODE(TDB_ERR_IO, -1);
}
return 0;
}

/* read a lump of data at a specified offset, maybe convert */
static int tdb_read(TDB_CONTEXT *tdb, tdb_off off, void *buf, tdb_len len, int cv)
{
if (tdb_oob(tdb, off + len, 0) != 0)
return -1;

if (tdb->map_ptr)
memcpy(buf, off + (char *)tdb->map_ptr, len);
#ifdef HAVE_PREAD
else if (pread(tdb->fd, buf, len, off) != (ssize_t)len) {
#else
else if (lseek(tdb->fd, off, SEEK_SET) != off
|| read(tdb->fd, buf, len) != (ssize_t)len) {
#endif
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_IO;
TDB_LOG((tdb, 0, "tdb_read failed at %d len=%d (%s)\n",
off, len, strerror(errno)));
return TDB_ERRCODE(TDB_ERR_IO, -1);
}
if (cv)
convert(buf, len);
return 0;
}

/* read a lump of data, allocating the space for it */
static char *tdb_alloc_read(TDB_CONTEXT *tdb, tdb_off offset, tdb_len len)
{
char *buf;

if (!(buf = malloc(len))) {
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_OOM;
TDB_LOG((tdb, 0, "tdb_alloc_read malloc failed len=%d (%s)\n",
len, strerror(errno)));
return TDB_ERRCODE(TDB_ERR_OOM, buf);
}
if (tdb_read(tdb, offset, buf, len, 0) == -1) {

```

```

SAFE_FREE(buf);
return NULL;
}
return buf;
}

/* read/write a tdb_off */
static int ofs_read(TDB_CONTEXT *tdb, tdb_off offset, tdb_off *d)
{
return tdb_read(tdb, offset, (char*)d, sizeof(*d), DOCONV());
}
static int ofs_write(TDB_CONTEXT *tdb, tdb_off offset, tdb_off *d)
{
tdb_off off = *d;
return tdb_write(tdb, offset, CONVERT(off), sizeof(*d));
}

/* read/write a record */
static int rec_read(TDB_CONTEXT *tdb, tdb_off offset, struct list_struct *rec)
{
if (tdb_read(tdb, offset, rec, sizeof(*rec), DOCONV()) == -1)
return -1;
if (TDB_BAD_MAGIC(rec)) {
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_CORRUPT;
TDB_LOG((tdb, 0, "rec_read bad magic 0x%x at offset=%d\n", rec->magic, offset));
return TDB_ERRCODE(TDB_ERR_CORRUPT, -1);
}
return tdb_oob(tdb, rec->next+sizeof(*rec), 0);
}
static int rec_write(TDB_CONTEXT *tdb, tdb_off offset, struct list_struct *rec)
{
struct list_struct r = *rec;
return tdb_write(tdb, offset, CONVERT(r), sizeof(r));
}

/* read a freelist record and check for simple errors */
static int rec_free_read(TDB_CONTEXT *tdb, tdb_off off, struct list_struct *rec)
{
if (tdb_read(tdb, off, rec, sizeof(*rec), DOCONV()) == -1)
return -1;

if (rec->magic == TDB_MAGIC) {
/* this happens when a app is shutdown while deleting a record - we should
not completely fail when this happens */
TDB_LOG((tdb, 0, "rec_free_read non-free magic 0x%x at offset=%d - fixing\n",
rec->magic, off));
rec->magic = TDB_FREE_MAGIC;
if (tdb_write(tdb, off, rec, sizeof(*rec)) == -1)
return -1;
}

if (rec->magic != TDB_FREE_MAGIC) {
/* Ensure ecode is set for log fn. */
tdb->ecode = TDB_ERR_CORRUPT;

```

```

TDB_LOG((tdb, 0, "rec_free_read bad magic 0x%x at offset=%d\n",
    rec->magic, off));
return TDB_ERRCODE(TDB_ERR_CORRUPT, -1);
}
if (tdb_oob(tdb, rec->next+sizeof(*rec), 0) != 0)
return -1;
return 0;
}

/* update a record tailer (must hold allocation lock) */
static int update_tailer(TDB_CONTEXT *tdb, tdb_off offset,
    const struct list_struct *rec)
{
    tdb_off totalsize;

    /* Offset of tailer from record header */
    totalsize = sizeof(*rec) + rec->rec_len;
    return ofs_write(tdb, offset + totalsize - sizeof(tdb_off),
        &totalsize);
}

static tdb_off tdb_dump_record(TDB_CONTEXT *tdb, tdb_off offset)
{
    struct list_struct rec;
    tdb_off tailer_ofs, tailer;

    if (tdb_read(tdb, offset, (char *)&rec, sizeof(rec), DOCONV()) == -1) {
        printf("ERROR: failed to read record at %u\n", offset);
        return 0;
    }

    printf(" rec: offset=%u next=%d rec_len=%d key_len=%d data_len=%d full_hash=0x%x magic=0x%x\n",
        offset, rec.next, rec.rec_len, rec.key_len, rec.data_len, rec.full_hash, rec.magic);

    tailer_ofs = offset + sizeof(rec) + rec.rec_len - sizeof(tdb_off);
    if (ofs_read(tdb, tailer_ofs, &tailer) == -1) {
        printf("ERROR: failed to read tailer at %u\n", tailer_ofs);
        return rec.next;
    }

    if (tailer != rec.rec_len + sizeof(rec)) {
        printf("ERROR: tailer does not match record! tailer=%u totalsize=%u\n",
            (unsigned)tailer, (unsigned)(rec.rec_len + sizeof(rec)));
    }
    return rec.next;
}

static int tdb_dump_chain(TDB_CONTEXT *tdb, int i)
{
    tdb_off rec_ptr, top;

    top = TDB_HASH_TOP(i);

    if (tdb_lock(tdb, i, F_WRLCK) != 0)

```

```

return -1;

if (ofs_read(tdb, top, &rec_ptr) == -1)
return tdb_unlock(tdb, i, F_WRLCK);

if (rec_ptr)
printf("hash=%d\n", i);

while (rec_ptr) {
rec_ptr = tdb_dump_record(tdb, rec_ptr);
}

return tdb_unlock(tdb, i, F_WRLCK);
}

void tdb_dump_all(TDB_CONTEXT *tdb)
{
int i;
for (i=0; i<tdb->header.hash_size; i++) {
tdb_dump_chain(tdb, i);
}
printf("freelist:\n");
tdb_dump_chain(tdb, -1);
}

int tdb_printfreelist(TDB_CONTEXT *tdb)
{
int ret;
long total_free = 0;
tdb_off offset, rec_ptr;
struct list_struct rec;

if ((ret = tdb_lock(tdb, -1, F_WRLCK)) != 0)
return ret;

offset = FREELIST_TOP;

/* read in the freelist top */
if (ofs_read(tdb, offset, &rec_ptr) == -1) {
tdb_unlock(tdb, -1, F_WRLCK);
return 0;
}

printf("freelist top=[0x%08x]\n", rec_ptr);
while (rec_ptr) {
if (tdb_read(tdb, rec_ptr, (char *)&rec, sizeof(rec), DOCONV()) == -1) {
tdb_unlock(tdb, -1, F_WRLCK);
return -1;
}

if (rec.magic != TDB_FREE_MAGIC) {
printf("bad magic 0x%08x in free list\n", rec.magic);
tdb_unlock(tdb, -1, F_WRLCK);
return -1;
}
}

```

```

printf("entry offset=[0x%08x], rec.rec_len = [0x%08x (%d)]\n", rec.next, rec.rec_len, rec.rec_len );
total_free += rec.rec_len;

/* move to the next record */
rec_ptr = rec.next;
}
printf("total rec_len = [0x%08x (%d)]\n", (int)total_free,
      (int)total_free);

return tdb_unlock(tdb, -1, F_WRLCK);
}

/* Remove an element from the freelist. Must have alloc lock. */
static int remove_from_freelist(TDB_CONTEXT *tdb, tdb_off off, tdb_off next)
{
    tdb_off last_ptr, i;

    /* read in the freelist top */
    last_ptr = FREELIST_TOP;
    while (ofs_read(tdb, last_ptr, &i) != -1 && i != 0) {
        if (i == off) {
            /* We've found it! */
            return ofs_write(tdb, last_ptr, &next);
        }
        /* Follow chain (next offset is at start of record) */
        last_ptr = i;
    }
    TDB_LOG((tdb, 0, "remove_from_freelist: not on list at off=%d\n", off));
    return TDB_ERRCODE(TDB_ERR_CORRUPT, -1);
}

/* Add an element into the freelist. Merge adjacent records if
   necessary. */
static int tdb_free(TDB_CONTEXT *tdb, tdb_off offset, struct list_struct *rec)
{
    tdb_off right, left;

    /* Allocation and tailer lock */
    if (tdb_lock(tdb, -1, F_WRLCK) != 0)
        return -1;

    /* set an initial tailer, so if we fail we don't leave a bogus record */
    if (update_tailer(tdb, offset, rec) != 0) {
        TDB_LOG((tdb, 0, "tdb_free: upfate_tailer failed!\n"));
        goto fail;
    }

    /* Look right first (I'm an Australian, dammit) */
    right = offset + sizeof(*rec) + rec->rec_len;
    if (right + sizeof(*rec) <= tdb->map_size) {
        struct list_struct r;

        if (tdb_read(tdb, right, &r, sizeof(r), DOCONV()) == -1) {

```

```

TDB_LOG((tdb, 0, "tdb_free: right read failed at %u\n", right));
goto left;
}

/* If it's free, expand to include it. */
if (r.magic == TDB_FREE_MAGIC) {
if (remove_from_freelist(tdb, right, r.next) == -1) {
TDB_LOG((tdb, 0, "tdb_free: right free failed at %u\n", right));
goto left;
}
rec->rec_len += sizeof(r) + r.rec_len;
}
}

left:
/* Look left */
left = offset - sizeof(tdb_off);
if (left > TDB_DATA_START(tdb->header.hash_size)) {
struct list_struct l;
tdb_off leftsize;

/* Read in tailer and jump back to header */
if (ofs_read(tdb, left, &leftsize) == -1) {
TDB_LOG((tdb, 0, "tdb_free: left offset read failed at %u\n", left));
goto update;
}
left = offset - leftsize;

/* Now read in record */
if (tdb_read(tdb, left, &l, sizeof(l), DOCONV()) == -1) {
TDB_LOG((tdb, 0, "tdb_free: left read failed at %u (%u)\n", left, leftsize));
goto update;
}

/* If it's free, expand to include it. */
if (l.magic == TDB_FREE_MAGIC) {
if (remove_from_freelist(tdb, left, l.next) == -1) {
TDB_LOG((tdb, 0, "tdb_free: left free failed at %u\n", left));
goto update;
} else {
offset = left;
rec->rec_len += leftsize;
}
}
}

update:
if (update_tailer(tdb, offset, rec) == -1) {
TDB_LOG((tdb, 0, "tdb_free: update_tailer failed at %u\n", offset));
goto fail;
}

/* Now, prepend to free list */
rec->magic = TDB_FREE_MAGIC;

```

```

if (ofs_read(tdb, FREELIST_TOP, &rec->next) == -1 ||
    rec_write(tdb, offset, rec) == -1 ||
    ofs_write(tdb, FREELIST_TOP, &offset) == -1) {
TDB_LOG((tdb, 0, "tdb_free record write failed at offset=%d\n", offset));
goto fail;
}

/* And we're done. */
tdb_unlock(tdb, -1, F_WRLCK);
return 0;

fail:
tdb_unlock(tdb, -1, F_WRLCK);
return -1;
}

/* expand a file. we prefer to use ftruncate, as that is what posix
   says to use for mmap expansion */
static int expand_file(TDB_CONTEXT *tdb, tdb_off size, tdb_off addition)
{
char buf[1024];
#ifdef HAVE_FTRUNCATE_EXTEND
if (ftruncate(tdb->fd, size+addition) != 0) {
TDB_LOG((tdb, 0, "expand_file ftruncate to %d failed (%s)\n",
    size+addition, strerror(errno)));
return -1;
}
#else
char b = 0;

#ifdef HAVE_PWRITE
if (pwrite(tdb->fd, &b, 1, (size+addition) - 1) != 1) {
#else
if (lseek(tdb->fd, (size+addition) - 1, SEEK_SET) != (size+addition) - 1 ||
    write(tdb->fd, &b, 1) != 1) {
#endif
TDB_LOG((tdb, 0, "expand_file to %d failed (%s)\n",
    size+addition, strerror(errno)));
return -1;
}
#endif
}

/* now fill the file with something. This ensures that the file isn't sparse, which would be
   very bad if we ran out of disk. This must be done with write, not via mmap */
memset(buf, 0x42, sizeof(buf));
while (addition) {
int n = addition > sizeof(buf) ? sizeof(buf) : addition;
#ifdef HAVE_PWRITE
int ret = pwrite(tdb->fd, buf, n, size);
#else
int ret;
if (lseek(tdb->fd, size, SEEK_SET) != size)
return -1;

```

```

ret = write(tdb->fd, buf, n);
#endif
if (ret != n) {
TDB_LOG((tdb, 0, "expand_file write of %d failed (%s)\n",
n, strerror(errno)));
return -1;
}
addition -= n;
size += n;
}
return 0;
}

/* expand the database at least size bytes by expanding the underlying
file and doing the mmap again if necessary */
static int tdb_expand(TDB_CONTEXT *tdb, tdb_off size)
{
struct list_struct rec;
tdb_off offset;

if (tdb_lock(tdb, -1, F_WRLCK) == -1) {
TDB_LOG((tdb, 0, "lock failed in tdb_expand\n"));
return -1;
}

/* must know about any previous expansions by another process */
tdb_oob(tdb, tdb->map_size + 1, 1);

/* always make room for at least 10 more records, and round
the database up to a multiple of TDB_PAGE_SIZE */
size = TDB_ALIGN(tdb->map_size + size*10, TDB_PAGE_SIZE) - tdb->map_size;

if (!(tdb->flags & TDB_INTERNAL))
tdb_munmap(tdb);

/*
* We must ensure the file is unmapped before doing this
* to ensure consistency with systems like OpenBSD where
* writes and mmaps are not consistent.
*/

/* expand the file itself */
if (!(tdb->flags & TDB_INTERNAL)) {
if (expand_file(tdb, tdb->map_size, size) != 0)
goto fail;
}

tdb->map_size += size;

if (tdb->flags & TDB_INTERNAL)
tdb->map_ptr = realloc(tdb->map_ptr, tdb->map_size);
else {
/*
* We must ensure the file is remapped before adding the space

```



```
* to ensure consistency with systems like OpenBSD where
* writes and mmaps are not consistent.
*/

/* We're ok if the mmap fails as we'll fallback to read/write */
tdb_mmap(tdb);
}

/* form a new freelist record */
memset(&rec, '\0', sizeof(rec));
rec.rec_len = size - sizeof(rec);

/* link it into the free list */
offset = tdb->map_size - size;
if (tdb_free(tdb, offset, &rec) == -1)
goto fail;

tdb_unlock(tdb, -1, F_WRLCK);
return 0;
fail:
tdb_unlock(tdb, -1, F_WRLCK);
return -1;
}

/* allocate some space from the free list. The offset returned points
to a unconnected list_struct within the database with room for at
least length bytes of total data

0 is returned if the space could not be allocated
*/
static tdb_off_t tdb_allocate(TDB_CONTEXT *tdb, tdb_len_t length,
struct list_struct *rec)
{
tdb_off_t rec_ptr, last_ptr, newrec_ptr;
struct list_struct newrec;

memset(&newrec, '\0', sizeof(newrec));

if (tdb_lock(tdb, -1, F_WRLCK) == -1)
return 0;

/* Extra bytes required for tailer */
length += sizeof(tdb_off_t);

again:
last_ptr = FREELIST_TOP;

/* read in the freelist top */
if (ofs_read(tdb, FREELIST_TOP, &rec_ptr) == -1)
goto fail;

/* keep looking until we find a freelist record big enough */
while (rec_ptr) {
if (rec_free_read(tdb, rec_ptr, rec) == -1)
```

```

goto fail;

if (rec->rec_len >= length) {
/* found it - now possibly split it up */
if (rec->rec_len > length + MIN_REC_SIZE) {
/* Length of left piece */
length = TDB_ALIGN(length, TDB_ALIGNMENT);

/* Right piece to go on free list */
newrec.rec_len = rec->rec_len
- (sizeof(*rec) + length);
newrec_ptr = rec_ptr + sizeof(*rec) + length;

/* And left record is shortened */
rec->rec_len = length;
} else
newrec_ptr = 0;

/* Remove allocated record from the free list */
if (ofs_write(tdb, last_ptr, &rec->next) == -1)
goto fail;

/* Update header: do this before we drop alloc
lock, otherwise tdb_free() might try to
merge with us, thinking we're free.
(Thanks Jeremy Allison). */
rec->magic = TDB_MAGIC;
if (rec_write(tdb, rec_ptr, rec) == -1)
goto fail;

/* Did we create new block? */
if (newrec_ptr) {
/* Update allocated record tailer (we
shortened it). */
if (update_tailer(tdb, rec_ptr, rec) == -1)
goto fail;

/* Free new record */
if (tdb_free(tdb, newrec_ptr, &newrec) == -1)
goto fail;
}

/* all done - return the new record offset */
tdb_unlock(tdb, -1, F_WRLCK);
return rec_ptr;
}

/* move to the next record */
last_ptr = rec_ptr;
rec_ptr = rec->next;
}

/* we didn't find enough space. See if we can expand the
database and if we can then try again */
if (tdb_expand(tdb, length + sizeof(*rec)) == 0)
goto again;
fail:

```

```

tdb_unlock(tdb, -1, F_WRLCK);
return 0;
}

/* initialise a new database with a specified hash size */
static int tdb_new_database(TDB_CONTEXT *tdb, int hash_size)
{
    struct tdb_header *newdb;
    int size, ret = -1;

    /* We make it up in memory, then write it out if not internal */
    size = sizeof(struct tdb_header) + (hash_size+1)*sizeof(tdb_off);
    if (!(newdb = calloc(size, 1)))
        return TDB_ERRCODE(TDB_ERR_OOM, -1);

    /* Fill in the header */
    newdb->version = TDB_VERSION;
    newdb->hash_size = hash_size;
    if (tdb->flags & TDB_INTERNAL) {
        tdb->map_size = size;
        tdb->map_ptr = (char *)newdb;
        memcpy(&tdb->header, newdb, sizeof(tdb->header));
        /* Convert the 'ondisk' version if asked. */
        CONVERT(*newdb);
        return 0;
    }
    if (lseek(tdb->fd, 0, SEEK_SET) == -1)
        goto fail;

    if (ftruncate(tdb->fd, 0) == -1)
        goto fail;

    /* This creates an endian-converted header, as if read from disk */
    CONVERT(*newdb);
    memcpy(&tdb->header, newdb, sizeof(tdb->header));
    /* Don't endian-convert the magic food! */
    memcpy(newdb->magic_food, TDB_MAGIC_FOOD, strlen(TDB_MAGIC_FOOD)+1);
    if (write(tdb->fd, newdb, size) != size)
        ret = -1;
    else
        ret = tdb_create_rwlock(tdb->fd, hash_size);

fail:
    SAFE_FREE(newdb);
    return ret;
}

/* Returns 0 on fail. On success, return offset of record, and fills
   in rec */
static tdb_off_tdb_find(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash,
    struct list_struct *r)
{
    tdb_off_tdb_rec_ptr;

```

```

/* read in the hash top */
if (ofs_read(tdb, TDB_HASH_TOP(hash), &rec_ptr) == -1)
return 0;

/* keep looking until we find the right record */
while (rec_ptr) {
if (rec_read(tdb, rec_ptr, r) == -1)
return 0;

if (!TDB_DEAD(r) && hash==r->full_hash && key.dsize==r->key_len) {
char *k;
/* a very likely hit - read the key */
k = tdb_alloc_read(tdb, rec_ptr + sizeof(*r),
r->key_len);
if (!k)
return 0;

if (memcmp(key.dptr, k, key.dsize) == 0) {
SAFE_FREE(k);
return rec_ptr;
}
SAFE_FREE(k);
}
rec_ptr = r->next;
}
return TDB_ERRCODE(TDB_ERR_NOEXIST, 0);
}

/* As tdb_find, but if you succeed, keep the lock */
static tdb_off_t tdb_find_lock_hash(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash, int locktype,
struct list_struct *rec)
{
u32 rec_ptr;

if (tdb_lock(tdb, BUCKET(hash), locktype) == -1)
return 0;
if (!(rec_ptr = tdb_find(tdb, key, hash, rec)))
tdb_unlock(tdb, BUCKET(hash), locktype);
return rec_ptr;
}

enum TDB_ERROR tdb_error(TDB_CONTEXT *tdb)
{
return tdb->ecode;
}

static struct tdb_errname {
enum TDB_ERROR ecode; const char *estring;
} emap[] = { {TDB_SUCCESS, "Success"},
{TDB_ERR_CORRUPT, "Corrupt database"},
{TDB_ERR_IO, "IO Error"},
{TDB_ERR_LOCK, "Locking error"},
{TDB_ERR_OOM, "Out of memory"},
{TDB_ERR_EXISTS, "Record exists"},
{TDB_ERR_NOLOCK, "Lock exists on other keys"},

```

```

    {TDB_ERR_NOEXIST, "Record does not exist"} };

/* Error string for the last tdb error */
const char *tdb_errorstr(TDB_CONTEXT *tdb)
{
    u32 i;
    for (i = 0; i < sizeof(emap) / sizeof(struct tdb_errname); i++)
        if (tdb->ecode == emap[i].ecode)
            return emap[i].estring;
    return "Invalid error code";
}

/* update an entry in place - this only works if the new data size
   is <= the old data size and the key exists.
   on failure return -1.
*/

static int tdb_update_hash(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash, TDB_DATA dbuf)
{
    struct list_struct rec;
    tdb_off rec_ptr;

    /* find entry */
    if (!(rec_ptr = tdb_find(tdb, key, hash, &rec)))
        return -1;

    /* must be long enough key, data and tailer */
    if (rec.rec_len < key.dsize + dbuf.dsize + sizeof(tdb_off)) {
        tdb->ecode = TDB_SUCCESS; /* Not really an error */
        return -1;
    }

    if (tdb_write(tdb, rec_ptr + sizeof(rec) + rec.key_len,
        dbuf.dptr, dbuf.dsize) == -1)
        return -1;

    if (dbuf.dsize != rec.data_len) {
        /* update size */
        rec.data_len = dbuf.dsize;
        return rec_write(tdb, rec_ptr, &rec);
    }

    return 0;
}

/* find an entry in the database given a key */
/* If an entry doesn't exist tdb_err will be set to
 * TDB_ERR_NOEXIST. If a key has no data attached
 * tdb_err will not be set. Both will return a
 * zero pptr and zero dsize.
*/

TDB_DATA tdb_fetch(TDB_CONTEXT *tdb, TDB_DATA key)
{

```

```

tdb_off rec_ptr;
struct list_struct rec;
TDB_DATA ret;
u32 hash;

/* find which hash bucket it is in */
hash = tdb->hash_fn(&key);
if (!(rec_ptr = tdb_find_lock_hash(tdb, key, hash, F_RDLCK, &rec)))
return tdb_null;

if (rec.data_len)
ret.dptr = tdb_alloc_read(tdb, rec_ptr + sizeof(rec) + rec.key_len,
    rec.data_len);
else
ret.dptr = NULL;
ret.dsize = rec.data_len;
tdb_unlock(tdb, BUCKET(rec.full_hash), F_RDLCK);
return ret;
}

/* check if an entry in the database exists

    note that 1 is returned if the key is found and 0 is returned if not found
    this doesn't match the conventions in the rest of this module, but is
    compatible with gdbm
*/
static int tdb_exists_hash(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash)
{
    struct list_struct rec;

    if (tdb_find_lock_hash(tdb, key, hash, F_RDLCK, &rec) == 0)
        return 0;
    tdb_unlock(tdb, BUCKET(rec.full_hash), F_RDLCK);
    return 1;
}

int tdb_exists(TDB_CONTEXT *tdb, TDB_DATA key)
{
    {
        u32 hash = tdb->hash_fn(&key);
        return tdb_exists_hash(tdb, key, hash);
    }
}

/* record lock stops delete underneath */
static int lock_record(TDB_CONTEXT *tdb, tdb_off off)
{
    {
        return off ? tdb_brlck(tdb, off, F_RDLCK, F_SETLKW, 0) : 0;
    }
}
/*
    Write locks override our own fcntl readlocks, so check it here.
    Note this is meant to be F_SETLK, *not* F_SETLKW, as it's not
    an error to fail to get the lock here.
*/

static int write_lock_record(TDB_CONTEXT *tdb, tdb_off off)
{
    {

```

```

struct tdb_traverse_lock *i;
for (i = &tdb->travlocks; i; i = i->next)
if (i->off == off)
return -1;
return tdb_brlck(tdb, off, F_WRLCK, F_SETLK, 1);
}

/*
Note this is meant to be F_SETLK, *not* F_SETLKW, as it's not
an error to fail to get the lock here.
*/

static int write_unlock_record(TDB_CONTEXT *tdb, tdb_off off)
{
return tdb_brlck(tdb, off, F_UNLCK, F_SETLK, 0);
}
/* fcntl locks don't stack: avoid unlocking someone else's */
static int unlock_record(TDB_CONTEXT *tdb, tdb_off off)
{
struct tdb_traverse_lock *i;
u32 count = 0;

if (off == 0)
return 0;
for (i = &tdb->travlocks; i; i = i->next)
if (i->off == off)
count++;
return (count == 1 ? tdb_brlck(tdb, off, F_UNLCK, F_SETLKW, 0) : 0);
}

/* actually delete an entry in the database given the offset */
static int do_delete(TDB_CONTEXT *tdb, tdb_off rec_ptr, struct list_struct*rec)
{
tdb_off last_ptr, i;
struct list_struct lastrec;

if (tdb->read_only) return -1;

if (write_lock_record(tdb, rec_ptr) == -1) {
/* Someone traversing here: mark it as dead */
rec->magic = TDB_DEAD_MAGIC;
return rec_write(tdb, rec_ptr, rec);
}
if (write_unlock_record(tdb, rec_ptr) != 0)
return -1;

/* find previous record in hash chain */
if (ofs_read(tdb, TDB_HASH_TOP(rec->full_hash), &i) == -1)
return -1;
for (last_ptr = 0; i != rec_ptr; last_ptr = i, i = lastrec.next)
if (rec_read(tdb, i, &lastrec) == -1)
return -1;

/* unlink it: next ptr is at start of record. */

```

```

if (last_ptr == 0)
last_ptr = TDB_HASH_TOP(rec->full_hash);
if (ofs_write(tdb, last_ptr, &rec->next) == -1)
return -1;

/* recover the space */
if (tdb_free(tdb, rec_ptr, rec) == -1)
return -1;
return 0;
}

/* Uses traverse lock: 0 = finish, -1 = error, other = record offset */
static int tdb_next_lock(TDB_CONTEXT *tdb, struct tdb_traverse_lock *tlock,
struct list_struct *rec)
{
int want_next = (tlock->off != 0);

/* Lock each chain from the start one. */
for (; tlock->hash < tdb->header.hash_size; tlock->hash++) {
if (tdb_lock(tdb, tlock->hash, F_WRLCK) == -1)
return -1;

/* No previous record? Start at top of chain. */
if (!tlock->off) {
if (ofs_read(tdb, TDB_HASH_TOP(tlock->hash),
&tlock->off) == -1)
goto fail;
} else {
/* Otherwise unlock the previous record. */
if (unlock_record(tdb, tlock->off) != 0)
goto fail;
}

if (want_next) {
/* We have offset of old record: grab next */
if (rec_read(tdb, tlock->off, rec) == -1)
goto fail;
tlock->off = rec->next;
}

/* Iterate through chain */
while( tlock->off) {
tdb_off current;
if (rec_read(tdb, tlock->off, rec) == -1)
goto fail;
if (!TDB_DEAD(rec)) {
/* Woohoo: we found one! */
if (lock_record(tdb, tlock->off) != 0)
goto fail;
return tlock->off;
}
/* Try to clean dead ones from old traverses */
current = tlock->off;
tlock->off = rec->next;
if (!tdb->read_only &&

```



```

        do_delete(tdb, current, rec) != 0)
goto fail;
}
tdb_unlock(tdb, tlock->hash, F_WRLCK);
want_next = 0;
}
/* We finished iteration without finding anything */
return TDB_ERRCODE(TDB_SUCCESS, 0);

fail:
tlock->off = 0;
if (tdb_unlock(tdb, tlock->hash, F_WRLCK) != 0)
TDB_LOG((tdb, 0, "tdb_next_lock: On error unlock failed!\n"));
return -1;
}

/* traverse the entire database - calling fn(tdb, key, data) on each element.
   return -1 on error or the record count traversed
   if fn is NULL then it is not called
   a non-zero return value from fn() indicates that the traversal should stop
   */
int tdb_traverse(TDB_CONTEXT *tdb, tdb_traverse_func fn, void *private)
{
    TDB_DATA key, dbuf;
    struct list_struct rec;
    struct tdb_traverse_lock tl = { NULL, 0, 0 };
    int ret, count = 0;

    /* This was in the initializaton, above, but the IRIX compiler
       * did not like it.  crh
       */
    tl.next = tdb->travlocks.next;

    /* fcntl locks don't stack: beware traverse inside traverse */
    tdb->travlocks.next = &tl;

    /* tdb_next_lock places locks on the record returned, and its chain */
    while ((ret = tdb_next_lock(tdb, &tl, &rec)) > 0) {
        count++;
        /* now read the full record */
        key.dptr = tdb_alloc_read(tdb, tl.off + sizeof(rec),
            rec.key_len + rec.data_len);
        if (!key.dptr) {
            ret = -1;
            if (tdb_unlock(tdb, tl.hash, F_WRLCK) != 0)
                goto out;
            if (unlock_record(tdb, tl.off) != 0)
                TDB_LOG((tdb, 0, "tdb_traverse: key.dptr == NULL and unlock_record failed!\n"));
            goto out;
        }
        key.dsize = rec.key_len;
        dbuf.dptr = key.dptr + rec.key_len;
        dbuf.dsize = rec.data_len;
    }

```

```

/* Drop chain lock, call out */
if (tdb_unlock(tdb, tl.hash, F_WRLCK) != 0) {
    ret = -1;
    goto out;
}
if (fn && fn(tdb, key, dbuf, private)) {
    /* They want us to terminate traversal */
    ret = count;
    if (unlock_record(tdb, tl.off) != 0) {
        TDB_LOG((tdb, 0, "tdb_traverse: unlock_record failed!\n"));
        ret = -1;
    }
    tdb->travlocks.next = tl.next;
    SAFE_FREE(key.dptr);
    return count;
}
SAFE_FREE(key.dptr);
}
out:
tdb->travlocks.next = tl.next;
if (ret < 0)
    return -1;
else
    return count;
}

/* find the first entry in the database and return its key */
TDB_DATA tdb_firstkey(TDB_CONTEXT *tdb)
{
    TDB_DATA key;
    struct list_struct rec;

    /* release any old lock */
    if (unlock_record(tdb, tdb->travlocks.off) != 0)
        return tdb_null;
    tdb->travlocks.off = tdb->travlocks.hash = 0;

    if (tdb_next_lock(tdb, &tdb->travlocks, &rec) <= 0)
        return tdb_null;
    /* now read the key */
    key.dsize = rec.key_len;
    key.dptr = tdb_alloc_read(tdb, tdb->travlocks.off + sizeof(rec), key.dsize);
    if (tdb_unlock(tdb, BUCKET(tdb->travlocks.hash), F_WRLCK) != 0)
        TDB_LOG((tdb, 0, "tdb_firstkey: error occurred while tdb_unlocking!\n"));
    return key;
}

/* find the next entry in the database, returning its key */
TDB_DATA tdb_nextkey(TDB_CONTEXT *tdb, TDB_DATA oldkey)
{
    u32 oldhash;
    TDB_DATA key = tdb_null;
    struct list_struct rec;
    char *k = NULL;

```

```

/* Is locked key the old key? If so, traverse will be reliable. */
if (tdb->travlocks.off) {
if (tdb_lock(tdb,tdb->travlocks.hash,F_WRLCK))
return tdb_null;
if (rec_read(tdb, tdb->travlocks.off, &rec) == -1
    || !(k = tdb_alloc_read(tdb,tdb->travlocks.off+sizeof(rec),
        rec.key_len))
    || memcmp(k, oldkey.dptr, oldkey.dsize) != 0) {
/* No, it wasn't: unlock it and start from scratch */
if (unlock_record(tdb, tdb->travlocks.off) != 0)
return tdb_null;
if (tdb_unlock(tdb, tdb->travlocks.hash, F_WRLCK) != 0)
return tdb_null;
tdb->travlocks.off = 0;
}

SAFE_FREE(k);
}

if (!tdb->travlocks.off) {
/* No previous element: do normal find, and lock record */
tdb->travlocks.off = tdb_find_lock_hash(tdb, oldkey, tdb->hash_fn(&oldkey), F_WRLCK, &rec);
if (!tdb->travlocks.off)
return tdb_null;
tdb->travlocks.hash = BUCKET(rec.full_hash);
if (lock_record(tdb, tdb->travlocks.off) != 0) {
TDB_LOG((tdb, 0, "tdb_nextkey: lock_record failed (%s)!\n", strerror(errno)));
return tdb_null;
}
}
oldhash = tdb->travlocks.hash;

/* Grab next record: locks chain and returned record,
   unlocks old record */
if (tdb_next_lock(tdb, &tdb->travlocks, &rec) > 0) {
key.dsize = rec.key_len;
key.dptr = tdb_alloc_read(tdb, tdb->travlocks.off+sizeof(rec),
    key.dsize);
/* Unlock the chain of this new record */
if (tdb_unlock(tdb, tdb->travlocks.hash, F_WRLCK) != 0)
TDB_LOG((tdb, 0, "tdb_nextkey: WARNING tdb_unlock failed!\n"));
}
/* Unlock the chain of old record */
if (tdb_unlock(tdb, BUCKET(oldhash), F_WRLCK) != 0)
TDB_LOG((tdb, 0, "tdb_nextkey: WARNING tdb_unlock failed!\n"));
return key;
}

/* delete an entry in the database given a key */
static int tdb_delete_hash(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash)
{
tdb_off rec_ptr;
struct list_struct rec;
int ret;

```

```

if (!(rec_ptr = tdb_find_lock_hash(tdb, key, hash, F_WRLCK, &rec)))
return -1;
ret = do_delete(tdb, rec_ptr, &rec);
if (tdb_unlock(tdb, BUCKET(rec.full_hash), F_WRLCK) != 0)
TDB_LOG((tdb, 0, "tdb_delete: WARNING tdb_unlock failed!\n"));
return ret;
}

int tdb_delete(TDB_CONTEXT *tdb, TDB_DATA key)
{
u32 hash = tdb->hash_fn(&key);
return tdb_delete_hash(tdb, key, hash);
}

/* store an element in the database, replacing any existing element
with the same key

return 0 on success, -1 on failure
*/
int tdb_store(TDB_CONTEXT *tdb, TDB_DATA key, TDB_DATA dbuf, int flag)
{
struct list_struct rec;
u32 hash;
tdb_off rec_ptr;
char *p = NULL;
int ret = 0;

/* find which hash bucket it is in */
hash = tdb->hash_fn(&key);
if (tdb_lock(tdb, BUCKET(hash), F_WRLCK) == -1)
return -1;

/* check for it existing, on insert. */
if (flag == TDB_INSERT) {
if (tdb_exists_hash(tdb, key, hash)) {
tdb->ecode = TDB_ERR_EXISTS;
goto fail;
}
} else {
/* first try in-place update, on modify or replace. */
if (tdb_update_hash(tdb, key, hash, dbuf) == 0)
goto out;
if (tdb->ecode == TDB_ERR_NOEXIST &&
flag == TDB_MODIFY) {
/* if the record doesn't exist and we are in TDB_MODIFY mode then
we should fail the store */
goto fail;
}
}

/* reset the error code potentially set by the tdb_update() */
tdb->ecode = TDB_SUCCESS;

/* delete any existing record - if it doesn't exist we don't
care. Doing this first reduces fragmentation, and avoids

```

```

        coalescing with `allocated' block before it's updated. */
if (flag != TDB_INSERT)
    tdb_delete_hash(tdb, key, hash);

/* Copy key+value *before* allocating free space in case malloc
   fails and we are left with a dead spot in the tdb. */

if (!(p = (char *)malloc(key.dsize + dbuf.dsize))) {
    tdb->ecode = TDB_ERR_OOM;
    goto fail;
}

memcpy(p, key.dptr, key.dsize);
if (dbuf.dsize)
    memcpy(p+key.dsize, dbuf.dptr, dbuf.dsize);

/* we have to allocate some space */
if (!(rec_ptr = tdb_allocate(tdb, key.dsize + dbuf.dsize, &rec)))
    goto fail;

/* Read hash top into next ptr */
if (ofs_read(tdb, TDB_HASH_TOP(hash), &rec.next) == -1)
    goto fail;

rec.key_len = key.dsize;
rec.data_len = dbuf.dsize;
rec.full_hash = hash;
rec.magic = TDB_MAGIC;

/* write out and point the top of the hash chain at it */
if (rec_write(tdb, rec_ptr, &rec) == -1
    || tdb_write(tdb, rec_ptr+sizeof(rec), p, key.dsize+dbuf.dsize) == -1
    || ofs_write(tdb, TDB_HASH_TOP(hash), &rec_ptr) == -1) {
    /* Need to tdb_unallocate() here */
    goto fail;
}
out:
SAFE_FREE(p);
tdb_unlock(tdb, BUCKET(hash), F_WRLCK);
return ret;
fail:
ret = -1;
goto out;
}

/* Attempt to append data to an entry in place - this only works if the new data size
   is <= the old data size and the key exists.
   on failure return -1. Record must be locked before calling.
*/
static int tdb_append_inplace(TDB_CONTEXT *tdb, TDB_DATA key, u32 hash, TDB_DATA new_dbuf)
{
    struct list_struct rec;
    tdb_off rec_ptr;

```

```

/* find entry */
if (!(rec_ptr = tdb_find(tdb, key, hash, &rec)))
return -1;

/* Append of 0 is always ok. */
if (new_dbuf.dsize == 0)
return 0;

/* must be long enough for key, old data + new data and tailer */
if (rec.rec_len < key.dsize + rec.data_len + new_dbuf.dsize + sizeof(tdb_off)) {
/* No room. */
tdb->ecode = TDB_SUCCESS; /* Not really an error */
return -1;
}

if (tdb_write(tdb, rec_ptr + sizeof(rec) + rec.key_len + rec.data_len,
new_dbuf.dptr, new_dbuf.dsize) == -1)
return -1;

/* update size */
rec.data_len += new_dbuf.dsize;
return rec_write(tdb, rec_ptr, &rec);
}

/* Append to an entry. Create if not exist. */

int tdb_append(TDB_CONTEXT *tdb, TDB_DATA key, TDB_DATA new_dbuf)
{
struct list_struct rec;
u32 hash;
tdb_off rec_ptr;
char *p = NULL;
int ret = 0;
size_t new_data_size = 0;

/* find which hash bucket it is in */
hash = tdb->hash_fn(&key);
if (tdb_lock(tdb, BUCKET(hash), F_WRLCK) == -1)
return -1;

/* first try in-place. */
if (tdb_append_inplace(tdb, key, hash, new_dbuf) == 0)
goto out;

/* reset the error code potentially set by the tdb_append_inplace() */
tdb->ecode = TDB_SUCCESS;

/* find entry */
if (!(rec_ptr = tdb_find(tdb, key, hash, &rec))) {
if (tdb->ecode != TDB_ERR_NOEXIST)
goto fail;

/* Not found - create. */

ret = tdb_store(tdb, key, new_dbuf, TDB_INSERT);

```

```
goto out;
}

new_data_size = rec.data_len + new_dbuf.dsize;

/* Copy key+old_value+value *before* allocating free space in case malloc
   fails and we are left with a dead spot in the tdb. */

if (!(p = (char *)malloc(key.dsize + new_data_size))) {
    tdb->ecode = TDB_ERR_OOM;
    goto fail;
}

/* Copy the key in place. */
memcpy(p, key.dptr, key.dsize);

/* Now read the old data into place. */
if (rec.data_len &&
    tdb_read(tdb, rec_ptr + sizeof(rec) + rec.key_len, p + key.dsize, rec.data_len, 0) == -1)
    goto fail;

/* Finally append the new data. */
if (new_dbuf.dsize)
    memcpy(p+key.dsize+rec.data_len, new_dbuf.dptr, new_dbuf.dsize);

/* delete any existing record - if it doesn't exist we don't
   care. Doing this first reduces fragmentation, and avoids
   coalescing with 'allocated' block before it's updated. */

tdb_delete_hash(tdb, key, hash);

if (!(rec_ptr = tdb_allocate(tdb, key.dsize + new_data_size, &rec)))
    goto fail;

/* Read hash top into next ptr */
if (ofs_read(tdb, TDB_HASH_TOP(hash), &rec.next) == -1)
    goto fail;

rec.key_len = key.dsize;
rec.data_len = new_data_size;
rec.full_hash = hash;
rec.magic = TDB_MAGIC;

/* write out and point the top of the hash chain at it */
if (rec_write(tdb, rec_ptr, &rec) == -1
    || tdb_write(tdb, rec_ptr+sizeof(rec), p, key.dsize+new_data_size) == -1
    || ofs_write(tdb, TDB_HASH_TOP(hash), &rec_ptr) == -1) {
    /* Need to tdb_unallocate() here */
    goto fail;
}

out:
SAFE_FREE(p);
tdb_unlock(tdb, BUCKET(hash), F_WRLCK);
```

```

return ret;

fail:
ret = -1;
goto out;
}

static int tdb_already_open(dev_t device,
    ino_t ino)
{
    TDB_CONTEXT *i;

    for (i = tdbbs; i; i = i->next) {
        if (i->device == device && i->inode == ino) {
            return 1;
        }
    }

    return 0;
}

/* This is based on the hash algorithm from gdbm */
static u32 default_tdb_hash(TDB_DATA *key)
{
    u32 value; /* Used to compute the hash value. */
    u32 i; /* Used to cycle through random values. */

    /* Set the initial value from the key size. */
    for (value = 0x238F13AF * key->dsiz, i=0; i < key->dsiz; i++)
        value = (value + (key->dp[i] << (i*5 % 24)));

    return (1103515243 * value + 12345);
}

/* open the database, creating it if necessary

The open_flags and mode are passed straight to the open call on the
database file. A flags value of O_WRONLY is invalid. The hash size
is advisory, use zero for a default value.

Return is NULL on error, in which case errno is also set. Don't
try to call tdb_error or tdb_errname, just do strerror(errno).

@param name may be NULL for internal databases. */
TDB_CONTEXT *tdb_open(const char *name, int hash_size, int tdb_flags,
    int open_flags, mode_t mode)
{
    return tdb_open_ex(name, hash_size, tdb_flags, open_flags, mode, NULL, NULL);
}

TDB_CONTEXT *tdb_open_ex(const char *name, int hash_size, int tdb_flags,
    int open_flags, mode_t mode,
    tdb_log_func log_fn,
    tdb_hash_func hash_fn)

```



```

{
TDB_CONTEXT *tdb;
struct stat st;
int rev = 0, locked = 0;
unsigned char *vp;
u32 veritest;

if (!(tdb = calloc(1, sizeof *tdb))) {
/* Can't log this */
errno = ENOMEM;
goto fail;
}
tdb->fd = -1;
tdb->name = NULL;
tdb->map_ptr = NULL;
tdb->flags = tdb_flags;
tdb->open_flags = open_flags;
tdb->log_fn = log_fn;
tdb->hash_fn = hash_fn ? hash_fn : default_tdb_hash;

if ((open_flags & O_ACCMODE) == O_WRONLY) {
TDB_LOG((tdb, 0, "tdb_open_ex: can't open tdb %s write-only\n",
name));
errno = EINVAL;
goto fail;
}

if (hash_size == 0)
hash_size = DEFAULT_HASH_SIZE;
if ((open_flags & O_ACCMODE) == O_RDONLY) {
tdb->read_only = 1;
/* read only databases don't do locking or clear if first */
tdb->flags |= TDB_NOLOCK;
tdb->flags &= ~TDB_CLEAR_IF_FIRST;
}

/* internal databases don't mmap or lock, and start off cleared */
if (tdb->flags & TDB_INTERNAL) {
tdb->flags |= (TDB_NOLOCK | TDB_NOMMAP);
tdb->flags &= ~TDB_CLEAR_IF_FIRST;
if (tdb_new_database(tdb, hash_size) != 0) {
TDB_LOG((tdb, 0, "tdb_open_ex: tdb_new_database failed!"));
goto fail;
}
goto internal;
}

if ((tdb->fd = open(name, open_flags, mode)) == -1) {
TDB_LOG((tdb, 5, "tdb_open_ex: could not open file %s: %s\n",
name, strerror(errno)));
goto fail; /* errno set by open(2) */
}

/* ensure there is only one process initialising at once */

```

```

if (tdb_brlck(tdb, GLOBAL_LOCK, F_WRLCK, F_SETLK, 0) == -1) {
    TDB_LOG((tdb, 0, "tdb_open_ex: failed to get global lock on %s: %s\n",
        name, strerror(errno)));
    goto fail; /* errno set by tdb_brlck */
}

/* we need to zero database if we are the only one with it open */
if ((tdb_flags & TDB_CLEAR_IF_FIRST) &&
    (locked = (tdb_brlck(tdb, ACTIVE_LOCK, F_WRLCK, F_SETLK, 0) == 0))) {
    open_flags |= O_CREAT;
    if (ftruncate(tdb->fd, 0) == -1) {
        TDB_LOG((tdb, 0, "tdb_open_ex: "
            "failed to truncate %s: %s\n",
            name, strerror(errno)));
        goto fail; /* errno set by ftruncate */
    }
}

if (read(tdb->fd, &tdb->header, sizeof(tdb->header)) != sizeof(tdb->header)
    || strcmp(tdb->header.magic_food, TDB_MAGIC_FOOD) != 0
    || (tdb->header.version != TDB_VERSION
    && !(rev = (tdb->header.version == TDB_BYTEREV(TDB_VERSION)))))) {
    /* its not a valid database - possibly initialise it */
    if (!(open_flags & O_CREAT) || tdb_new_database(tdb, hash_size) == -1) {
        errno = EIO; /* ie bad format or something */
        goto fail;
    }
    rev = (tdb->flags & TDB_CONVERT);
}
vp = (unsigned char *)&tdb->header.version;
vertest = (((u32)vp[0]) << 24) | (((u32)vp[1]) << 16) |
    (((u32)vp[2]) << 8) | (u32)vp[3];
tdb->flags |= (vertest == TDB_VERSION) ? TDB_BIGENDIAN : 0;
if (!rev)
    tdb->flags &= ~TDB_CONVERT;
else {
    tdb->flags |= TDB_CONVERT;
    convert(&tdb->header, sizeof(tdb->header));
}
if (fstat(tdb->fd, &st) == -1)
    goto fail;

/* Is it already in the open list? If so, fail. */
if (tdb_already_open(st.st_dev, st.st_ino)) {
    TDB_LOG((tdb, 2, "tdb_open_ex: "
        "%s (%d,%d) is already open in this process\n",
        name, (int)st.st_dev, (int)st.st_ino));
    errno = EBUSY;
    goto fail;
}

if (!(tdb->name = (char *)strdup(name))) {
    errno = ENOMEM;
    goto fail;
}

```

```

tdb->map_size = st.st_size;
tdb->device = st.st_dev;
tdb->inode = st.st_ino;
tdb->locked = calloc(tdb->header.hash_size+1, sizeof(tdb->locked[0]));
if (!tdb->locked) {
    TDB_LOG((tdb, 2, "tdb_open_ex: "
        "failed to allocate lock structure for %s\n",
        name));
    errno = ENOMEM;
    goto fail;
}
tdb_mmap(tdb);
if (locked) {
    if (!tdb->read_only)
        if (tdb_clear_spinlocks(tdb) != 0) {
            TDB_LOG((tdb, 0, "tdb_open_ex: "
                "failed to clear spinlock\n"));
            goto fail;
        }
        if (tdb_brlock(tdb, ACTIVE_LOCK, F_UNLCK, F_SETLK, 0) == -1) {
            TDB_LOG((tdb, 0, "tdb_open_ex: "
                "failed to take ACTIVE_LOCK on %s: %s\n",
                name, strerror(errno)));
            goto fail;
        }
    }

    /* We always need to do this if the CLEAR_IF_FIRST flag is set, even if
       we didn't get the initial exclusive lock as we need to let all other
       users know we're using it. */

    if (tdb_flags & TDB_CLEAR_IF_FIRST) {
        /* leave this lock in place to indicate it's in use */
        if (tdb_brlock(tdb, ACTIVE_LOCK, F_RDLCK, F_SETLKW, 0) == -1)
            goto fail;
    }

    internal:
    /* Internal (memory-only) databases skip all the code above to
       * do with disk files, and resume here by releasing their
       * global lock and hooking into the active list. */
    if (tdb_brlock(tdb, GLOBAL_LOCK, F_UNLCK, F_SETLKW, 0) == -1)
        goto fail;
    tdb->next = tdb;
    tdb = tdb;
    return tdb;

fail:
{ int save_errno = errno;

if (!tdb)

```

```

return NULL;

if (tdb->map_ptr) {
if (tdb->flags & TDB_INTERNAL)
SAFE_FREE(tdb->map_ptr);
else
tdb_munmap(tdb);
}
SAFE_FREE(tdb->name);
if (tdb->fd != -1)
if (close(tdb->fd) != 0)
TDB_LOG((tdb, 5, "tdb_open_ex: failed to close tdb->fd on error!\n"));
SAFE_FREE(tdb->locked);
SAFE_FREE(tdb);
errno = save_errno;
return NULL;
}
}

/**
 * Close a database.
 *
 * @returns -1 for error; 0 for success.
 */
int tdb_close(TDB_CONTEXT *tdb)
{
TDB_CONTEXT **i;
int ret = 0;

if (tdb->map_ptr) {
if (tdb->flags & TDB_INTERNAL)
SAFE_FREE(tdb->map_ptr);
else
tdb_munmap(tdb);
}
SAFE_FREE(tdb->name);
if (tdb->fd != -1)
ret = close(tdb->fd);
SAFE_FREE(tdb->locked);

/* Remove from contexts list */
for (i = &tdbs; *i; i = &(*i)->next) {
if (*i == tdb) {
*i = tdb->next;
break;
}
}

memset(tdb, 0, sizeof(*tdb));
SAFE_FREE(tdb);

return ret;
}

/* lock/unlock entire database */

```

```
int tdb_lockall(TDB_CONTEXT *tdb)
{
    u32 i;

    /* There are no locks on read-only dbs */
    if (tdb->read_only)
        return TDB_ERRCODE(TDB_ERR_LOCK, -1);
    for (i = 0; i < tdb->header.hash_size; i++)
        if (tdb_lock(tdb, i, F_WRLCK))
            break;

    /* If error, release locks we have... */
    if (i < tdb->header.hash_size) {
        u32 j;

        for (j = 0; j < i; j++)
            tdb_unlock(tdb, j, F_WRLCK);
        return TDB_ERRCODE(TDB_ERR_NOLOCK, -1);
    }

    return 0;
}

void tdb_unlockall(TDB_CONTEXT *tdb)
{
    u32 i;
    for (i=0; i < tdb->header.hash_size; i++)
        tdb_unlock(tdb, i, F_WRLCK);
}

/* lock/unlock one hash chain. This is meant to be used to reduce
   contention - it cannot guarantee how many records will be locked */
int tdb_chainlock(TDB_CONTEXT *tdb, TDB_DATA key)
{
    return tdb_lock(tdb, BUCKET(tdb->hash_fn(&key)), F_WRLCK);
}

int tdb_chainunlock(TDB_CONTEXT *tdb, TDB_DATA key)
{
    return tdb_unlock(tdb, BUCKET(tdb->hash_fn(&key)), F_WRLCK);
}

int tdb_chainlock_read(TDB_CONTEXT *tdb, TDB_DATA key)
{
    return tdb_lock(tdb, BUCKET(tdb->hash_fn(&key)), F_RDLCK);
}

int tdb_chainunlock_read(TDB_CONTEXT *tdb, TDB_DATA key)
{
    return tdb_unlock(tdb, BUCKET(tdb->hash_fn(&key)), F_RDLCK);
}

/* register a logging function */
void tdb_logging_function(TDB_CONTEXT *tdb, void (*fn)(TDB_CONTEXT *, int, const char *, ...))
```

```

{
tdb->log_fn = fn;
}

/* reopen a tdb - this can be used after a fork to ensure that we have an independent
seek pointer from our parent and to re-establish locks */
int tdb_reopen(TDB_CONTEXT *tdb)
{
struct stat st;

if (tdb->flags & TDB_INTERNAL)
return 0; /* Nothing to do. */
if (tdb_munmap(tdb) != 0) {
TDB_LOG((tdb, 0, "tdb_reopen: munmap failed (%s)\n", strerror(errno)));
goto fail;
}
if (close(tdb->fd) != 0)
TDB_LOG((tdb, 0, "tdb_reopen: WARNING closing tdb->fd failed!\n"));
tdb->fd = open(tdb->name, tdb->open_flags & ~(O_CREAT|O_TRUNC), 0);
if (tdb->fd == -1) {
TDB_LOG((tdb, 0, "tdb_reopen: open failed (%s)\n", strerror(errno)));
goto fail;
}
if (fstat(tdb->fd, &st) != 0) {
TDB_LOG((tdb, 0, "tdb_reopen: fstat failed (%s)\n", strerror(errno)));
goto fail;
}
if (st.st_ino != tdb->inode || st.st_dev != tdb->device) {
TDB_LOG((tdb, 0, "tdb_reopen: file dev/inode has changed!\n"));
goto fail;
}
tdb_mmap(tdb);
if ((tdb->flags & TDB_CLEAR_IF_FIRST) && (tdb_brlock(tdb, ACTIVE_LOCK, F_RDLCK, F_SETLKW, 0)
== -1)) {
TDB_LOG((tdb, 0, "tdb_reopen: failed to obtain active lock\n"));
goto fail;
}

return 0;

fail:
tdb_close(tdb);
return -1;
}

/* reopen all tdb's */
int tdb_reopen_all(void)
{
TDB_CONTEXT *tdb;

for (tdb=tdbs; tdb; tdb = tdb->next) {
/* Ensure no clear-if-first. */
tdb->flags &= ~TDB_CLEAR_IF_FIRST;
if (tdb_reopen(tdb) != 0)
return -1;
}
}

```

```
}  
  
return 0;  
}
```

1.113 procps

1.113.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LIBRARY GENERAL PUBLIC LICENSE Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis

or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this

will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy,

and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do

this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6.

Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot

use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to

refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if

necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.114 psmisc

1.114.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

```
This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.115 quagga

1.115.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this

will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do

this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may

distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot

use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then

the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.116 radvd

1.116.1 Available under license

The author(s) grant permission for redistribution and use in source and binary forms, with or without modification, of the software and documentation provided that the following conditions are met:

0. If you receive a version of the software that is specifically labelled as not being for redistribution (check the version message and/or README), you are not permitted to redistribute that version of the software in any way or form.
1. All terms of all other applicable copyrights and licenses must be followed.
2. Redistributions of source code must retain the authors' copyright notice(s), this list of conditions, and the following disclaimer.
3. Redistributions in binary form must reproduce the authors' copyright notice(s), this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
4. All advertising materials mentioning features or use of this software must display the following acknowledgement with the name(s) of the authors as specified in the copyright notice(s) substituted where indicated:

This product includes software developed by the authors which are mentioned at the start of the source files and other contributors.

5. Neither the name(s) of the author(s) nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ITS AUTHORS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.117 readline

1.117.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.118 rpcbind

1.118.1 Available under license

```
/*
 * Copyright (c) Copyright (c) Bull S.A. 2005 All Rights Reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. The name of the author may not be used to endorse or promote products
 * derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR
 * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 * IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
 * THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */
/*
 * Copyright (c) 2009, Sun Microsystems, Inc.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 * - Redistributions of source code must retain the above copyright notice,
 * this list of conditions and the following disclaimer.
 * - Redistributions in binary form must reproduce the above copyright notice,
 * this list of conditions and the following disclaimer in the documentation
 * and/or other materials provided with the distribution.
 * - Neither the name of Sun Microsystems, Inc. nor the names of its
 * contributors may be used to endorse or promote products derived
 * from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
```

```

* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*/
/*
* Copyright (c) 1986 - 1991 by Sun Microsystems, Inc.
*/
/*
* rpcinfo: ping a particular rpc program
* or dump the the registered programs on the remote machine.
*/

/*
* We are for now defining PORTMAP here. It doesnt even compile
* unless it is defined.
*/
#ifndef PORTMAP
#define PORTMAP
#endif

/*
* If PORTMAP is defined, rpcinfo will talk to both portmapper and
* rpcbind programs; else it talks only to rpcbind. In the latter case
* all the portmapper specific options such as -u, -t, -p become void.
*/
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <rpc/rpc.h>
#include <stdio.h>
#include <rpc/rpcb_prot.h>
#include <rpc/rpcent.h>
#include <rpc/nettype.h>
#include <rpc/rpc_com.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <err.h>
#include <ctype.h>

#ifdef PORTMAP/* Support for version 2 portmapper */
#include <netinet/in.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <rpc/pmap_prot.h>
#include <rpc/pmap_clnt.h>
#endif

#define max(a,b) ((a) > (b) ? (a) : (b))

#define MIN_VERS((u_long)0)
#define MAX_VERS((u_long)4294967295UL)
#define UNKNOWN"unknown"

/*

```

```

* Functions to be performed.
*/
#defineNONE0/* no function */
#definePMAPDUMP1/* dump portmapper registrations */
#defineTCPPING2/* ping TCP service */
#defineUDPPING3/* ping UDP service */
#defineBROADCAST4/* ping broadcast service */
#defineDELETES5/* delete registration for the service */
#defineADDRPING6/* pings at the given address */
#definePROGPING7/* pings a program on a given host */
#defineRPCBDUMP8/* dump rpcbind registrations */
#defineRPCBDUMP_SHORT9/* dump rpcbind registrations - short version */
#defineRPCBADDRLIST10/* dump addr list about one prog */
#defineRPCBGETSTAT11/* Get statistics */

struct netidlist
{
    char *netid;
    struct netidlist *next;
};

struct verslist
{
    int vers;
    struct verslist *next;
};

struct rpcbdump_short
{
    u_long prog;
    struct verslist *vlist;
    struct netidlist *nlist;
    struct rpcbdump_short *next;
    char *owner;
};

#ifdef PORTMAP
static void ip_ping (u_short, char *, int, char **);
static void pmapdump (int, char **);
static CLIENT *ip_getclient(const char *hostname, rpcprog_t prognum, rpcvers_t versnum, const char *proto);
#endif

static bool_t reply_proc (void *, struct netbuf *, struct netconfig *);
static void brdest (int, char **);
static void addrping (char *, char *, int, char **);
static void progping (char *, int, char **);
static CLIENT *clnt_addr_create (char *, struct netconfig *, u_long, u_long);
static CLIENT *clnt_rpcbind_create (char *, int, struct netbuf **);
static CLIENT *getclnthandle (char *, struct netconfig *, u_long,
    struct netbuf **);
static int pstatus (CLIENT *, u_long, u_long);
static void rpcbdump (int, char *, int, char **);
static void rpcbgetstat (int, char **);

```

```

static void rpcbaddrlist (char *, int, char **);
static void deletereg (char *, int, char **);
static void print_rmtcallstat (int, rpcb_stat *);
static void print_getaddrstat (int, rpcb_stat *);
static void usage (void);
static u_long getprognum (char *);
static u_long getvers (char *);
static char *spaces (int);
static bool_t add_version (struct rpcbdump_short *, u_long);
static bool_t add_netid (struct rpcbdump_short *, char *);

```

```

int main (int argc, char **argv);

```

```

int
main (int argc, char **argv)
{
    register int c;
    int errflg;
    int function;
    char *netid = NULL;
    char *address = NULL;
#ifdef PORTMAP
    char *strptr;
    u_short portnum = 0;
#endif

    function = NONE;
    errflg = 0;
#ifdef PORTMAP
    while ((c = getopt (argc, argv, "a:bdlmn:psT:u")) != -1)
#else
    while ((c = getopt (argc, argv, "a:bdlmn:sT:")) != -1)
#endif
    {
        switch (c)
        {
#ifdef PORTMAP
        case 'p':
            if (function != NONE)
                errflg = 1;
            else
                function = PMAPDUMP;
            break;

        case 't':
            if (function != NONE)
                errflg = 1;
            else
                function = TCPPING;
            break;

        case 'u':
            if (function != NONE)
                errflg = 1;
            else

```



```
        function = UDPPING;
        break;

    case 'n':
        portnum = (u_short) strtol (optarg, &strptr, 10);
        if (strptr == optarg || *strptr != '\0')
        {
            fprintf (stderr, "rpcinfo: %s is illegal port number\n",
                optarg);
            exit (1);
        }
        break;
#endif
```

```
    case 'a':
        address = optarg;
        if (function != NONE)
            errflg = 1;
        else
            function = ADDRPING;
        break;
```

```
    case 'b':
        if (function != NONE)
            errflg = 1;
        else
            function = BROADCAST;
        break;
```

```
    case 'd':
        if (function != NONE)
            errflg = 1;
        else
            function = DELETES;
        break;
```

```
    case 'l':
        if (function != NONE)
            errflg = 1;
        else
            function = RPCBADDRLIST;
        break;
```

```
    case 'm':
        if (function != NONE)
            errflg = 1;
        else
            function = RPCBGETSTAT;
        break;
```

```
    case 's':
        if (function != NONE)
            errflg = 1;
        else
```

```

        function = RPCBDUMP_SHORT;
        break;

    case 'T':
        netid = optarg;
        break;
    case '?':
        errflg = 1;
        break;
    }
}

if (errflg || ((function == ADDRPING) && !netid))
{
    usage ();
    return 1;
}

if (function == NONE)
{
    if (argc - optind > 1)
        function = PROGPING;
    else
        function = RPCBDUMP;
}

switch (function)
{
#ifdef PORTMAP
    case PMAPDUMP:
        if (portnum != 0)
        {
            usage ();
            return 1;
        }
        pmapdump (argc - optind, argv + optind);
        break;

    case UDPPING:
        ip_ping (portnum, "udp", argc - optind, argv + optind);
        break;

    case TCPPING:
        ip_ping (portnum, "tcp", argc - optind, argv + optind);
        break;
#endif
    case BROADCAST:
        brdcst (argc - optind, argv + optind);
        break;
    case DELETES:
        deletereg (netid, argc - optind, argv + optind);
        break;
    case ADDRPING:
        addrping (address, netid, argc - optind, argv + optind);
        break;
}

```

```

    case PROGPING:
        progping (netid, argc - optind, argv + optind);
        break;
    case RPCBDUMP:
    case RPCBDUMP_SHORT:
        rpcbdump (function, netid, argc - optind, argv + optind);
        break;
    case RPCBGETSTAT:
        rpcbgetstat (argc - optind, argv + optind);
        break;
    case RPCBADDRLIST:
        rpcbaddrlist (netid, argc - optind, argv + optind);
        break;
    }
    return (0);
}

static CLIENT *
local_rpcb (rpcprog_t prog, rpcvers_t vers)
{
    #if 0
        void *localhandle;
        struct netconfig *nconf;
        CLIENT *clnt;

        localhandle = setnetconfig();
        while ((nconf = getnetconfig(localhandle)) != NULL) {
            if (nconf->nc_protofmly != NULL &&
                strcmp(nconf->nc_protofmly, NC_LOOPBACK) == 0)
                break;
        }
        if (nconf == NULL) {
            warnx("getnetconfig: %s", nc_serror());
            return (NULL);
        }

        clnt = clnt_tp_create ("localhost"/ NULL, prog, vers, nconf);
        endnetconfig(localhandle);
        return clnt;
    #else
        struct netbuf nbuf;
        struct sockaddr_un sun;
        int sock;

        memset (&sun, 0, sizeof sun);
        sock = socket (AF_LOCAL, SOCK_STREAM, 0);
        if (sock < 0)
            return NULL;

        sun.sun_family = AF_LOCAL;
        strcpy (sun.sun_path, _PATH_RPCBINDSOCK);
        nbuf.len = SUN_LEN (&sun);
        nbuf.maxlen = sizeof (struct sockaddr_un);
        nbuf.buf = &sun;
    #endif
}

```

```

    return clnt_vc_create (sock, &nbuf, prog, vers, 0, 0);
#endif
}

#ifndef PORTMAP
static enum clnt_stat
ip_ping_one(client, vers)
    CLIENT *client;
    u_int32_t vers;
{
    struct timeval to = { .tv_sec = 10, .tv_usec = 0 };

    (void) CLNT_CONTROL (client, CLSET_VERS, &vers);
    return CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
        (char *) NULL, (xdrproc_t) xdr_void, (char *) NULL,
        to);
}

/*
 * If portnum is 0, then go and get the address from portmapper, which happens
 * transparently through clnt*_create(); If version number is not given, it
 * tries to find out the version number by making a call to version 0 and if
 * that fails, it obtains the high order and the low order version number. If
 * version 0 calls succeeds, it tries for MAXVERS call and repeats the same.
 */
static void
ip_ping (portnum, proto, argc, argv)
    u_short portnum;
    char *proto;
    int argc;
    char **argv;
{
    CLIENT *client;
    enum clnt_stat rpc_stat;
    const char *hostname;
    u_long prognum, vers, minvers, maxvers;
    struct rpc_err rpcerr;
    int failure = 0;

    if (argc < 2 || argc > 3)
    {
        usage ();
        exit (1);
    }

    hostname = argv[0];
    prognum = getprognum (argv[1]);
    if (argc == 2)
    {
        /* Version number not known */
        /*
         * A call to version 0 should fail with a program/version
         * mismatch, and give us the range of versions supported.
         */
        vers = MIN_VERS;
    }

```

```
    }
else
    {
        vers = getvers (argv[2]);
    }

client = ip_getclient(hostname, prognum, vers, proto);

rpc_stat = ip_ping_one(client, vers);
if (argc != 2)
    {
        /* Version number was known */
        if (pstatus (client, prognum, vers) < 0)
            exit (1);
        (void) CLNT_DESTROY (client);
        return;
    }

/* Version number not known */
if (rpc_stat == RPC_PROGVERSMISMATCH)
    {
        clnt_geterr (client, &rpcerr);
        minvers = rpcerr.re_vers.low;
        maxvers = rpcerr.re_vers.high;
    }
else if (rpc_stat == RPC_SUCCESS)
    {
        /*
         * Oh dear, it DOES support version 0.
         * Let's try version MAX_VERS.
         */
        rpc_stat = ip_ping_one(client, MAX_VERS);
        if (rpc_stat == RPC_PROGVERSMISMATCH)
        {
            clnt_geterr (client, &rpcerr);
            minvers = rpcerr.re_vers.low;
            maxvers = rpcerr.re_vers.high;
        }
        else if (rpc_stat == RPC_SUCCESS)
        {
            /*
             * It also supports version MAX_VERS.
             * Looks like we have a wise guy.
             * OK, we give them information on all
             * 4 billion versions they support...
             */
            minvers = 0;
            maxvers = MAX_VERS;
        }
    }
else
    {
        (void) pstatus (client, prognum, MAX_VERS);
        exit (1);
    }
}
```

```

    }
    else
    {
        (void) pstatus (client, prognum, (u_long) 0);
        exit (1);
    }
    for (vers = minvers; vers <= maxvers; vers++)
    {
        rpc_stat = ip_ping_one(client, vers);
        if (pstatus (client, prognum, vers) < 0)
            failure = 1;
    }
    if (failure)
        exit (1);
    (void) CLNT_DESTROY (client);
    return;
}

/*
 * Dump all the portmapper registrations
 */
static void
pmapdump (argc, argv)
    int argc;
    char **argv;
{
    struct pmaplist *head = NULL;
    struct timeval minutetimeout;
    register CLIENT *client;
    struct rpcent *rpc;
    enum clnt_stat clnt_st;
    struct rpc_err err;
    char *host = NULL;

    if (argc > 1)
    {
        usage ();
        exit (1);
    }
    if (argc == 1)
    {
        host = argv[0];

        /* This is a little bit more complicated than it should be.
         * ip_getclient will do an rpcb_getaddr call to identify the
         * port of the portmapper - but it works, and it's easier than
         * creating a copy of ip_getclient that avoids the getaddr call.
         */
        client = ip_getclient(host, PMAPPROG, PMAPVERS, "tcp");
    }
    else
        client = local_rpcb (PMAPPROG, PMAPVERS);

    if (client == NULL)
    {

```

```

    if (rpc_createerr.cf_stat == RPC_TLIERROR)
    {
        /*
         * "Misc. TLI error" is not too helpful. Most likely
         * the connection to the remote server timed out, so
         * this error is at least less perplexing.
         */
        rpc_createerr.cf_stat = RPC_PMAPFAILURE;
        rpc_createerr.cf_error.re_status = RPC_FAILED;
    }
    clnt_pcreateerror ("rpcinfo: can't contact portmapper");
    exit (1);
}

minutetimeout.tv_sec = 60;
minutetimeout.tv_usec = 0;

clnt_st = CLNT_CALL (client, PMAPPROC_DUMP, (xdrproc_t) xdr_void,
    NULL, (xdrproc_t) xdr_pmaplist_ptr, (char *) &head,
    minutetimeout);
if (clnt_st != RPC_SUCCESS)
{
    if ((clnt_st == RPC_PROGVERSMISMATCH) || (clnt_st == RPC_PROGUNAVAIL))
    {
        CLNT_GETERR (client, &err);
        if (err.re_vers.low > PMAPVERS)
            fprintf (stderr,
                "%s does not support portmapper. Try rpcinfo %s instead\n",
                host, host);
        exit (1);
    }
    clnt_perror (client, "rpcinfo: can't contact portmapper");
    exit (1);
}
if (head == NULL)
{
    printf ("No remote programs registered.\n");
}
else
{
    printf ("  program vers proto  port  service\n");
    for (; head != NULL; head = head->pml_next)
    {
        printf ("%10ld%5ld", head->pml_map.pm_prog, head->pml_map.pm_vers);
        if (head->pml_map.pm_prot == IPPROTO_UDP)
            printf ("%6s", "udp");
        else if (head->pml_map.pm_prot == IPPROTO_TCP)
            printf ("%6s", "tcp");
        else
            printf ("%6ld", head->pml_map.pm_prot);
        printf ("%7ld", head->pml_map.pm_port);
        rpc = getrpcbyname (head->pml_map.pm_prog);
        if (rpc)
            printf (" %s\n", rpc->r_name);
    }
}

```

```

    else
        printf("\n");
    }
}

/*
 * Try to obtain the address of a given host/program/version, using the
 * specified protocol (one of udp or tcp).
 * This loops over all netconfig entries (according to the order given by
 * netpath and the config file), and tries to resolve the hostname, and obtain
 * the address using rpcb_getaddr.
 */
CLIENT *
ip_getclient(hostname, prognum, versnum, proto)
    const char *hostname;
    rpcprog_t prognum;
    rpcvers_t versnum;
    const char *proto;
{
    void *handle;
    enum clnt_stat saved_stat = RPC_SUCCESS;
    struct netconfig *nconf, *result = NULL;
    struct netbuf bind_address;
    struct sockaddr_storage __sa;
    CLIENT *client;

    memset(&bind_address, 0, sizeof(bind_address));
    bind_address.maxlen = sizeof(__sa);
    bind_address.buf = &__sa;

    handle = setnetconfig();
    while ((nconf = getnetconfig(handle)) != NULL)
    {
        if (!strcmp(nconf->nc_proto, proto)) {
            if (rpcb_getaddr(prognum, versnum, nconf, &bind_address, hostname))
            {
                result = getnetconfigent(nconf->nc_netid);
                endnetconfig(handle);
                break;
            }
        }

        if (rpc_createerr.cf_stat != RPC_UNKNOWNHOST)
        {
            clnt_pcreateerror (hostname);
            exit (1);
        }
    }

    saved_stat = rpc_createerr.cf_stat;
}

if (result == NULL)
{
    if (saved_stat != RPC_SUCCESS)

```



```

    {
        rpc_createerr.cf_stat = saved_stat;
        clnt_pcreateerror (hostname);
    }
    else
        fprintf (stderr, "Cannot find suitable transport for protocol %s\n", proto);

    exit (1);
}

client = clnt_tli_create(RPC_ANYFD, result, &bind_address, prognum, versnum, 0, 0);
if (client == NULL)
{
    clnt_pcreateerror(hostname);
    exit (1);
}

freenetconfigt(result);
return client;
}
#endif /* PORTMAP */

static int
sa_len(struct sockaddr *sa)
{
    socklen_t salen;

    switch (sa->sa_family)
    {
        case AF_LOCAL:
            salen = sizeof (struct sockaddr_un);
            break;
        case AF_INET:
            salen = sizeof (struct sockaddr_in);
            break;
        case AF_INET6:
            salen = sizeof (struct sockaddr_in6);
            break;
        default:
            salen = 0;
            break;
    }
    return salen;
}

/*
 * reply_proc collects replies from the broadcast.
 * to get a unique list of responses the output of rpcinfo should
 * be piped through sort(1) and then uniq(1).
 */

/*ARGSUSED*/ static bool_t
reply_proc (res, who, nconf)
    void *res; /* Nothing comes back */

```

```

    struct netbuf *who; /* Who sent us the reply */
    struct netconfig *nconf; /* On which transport the reply came */
{
    char *uaddr;
    char hostbuf[NI_MAXHOST];
    char *hostname;
    struct sockaddr *sa = (struct sockaddr *) who->buf;

    if (getnameinfo (sa, sa_len (sa), hostbuf, NI_MAXHOST, NULL, 0, 0))
    {
        hostname = UNKNOWN;
    }
    else
    {
        hostname = hostbuf;
    }
    if (!(uaddr = taddr2uaddr (nconf, who)))
    {
        uaddr = UNKNOWN;
    }
    printf ("%s\t%s\n", uaddr, hostname);
    if (strcmp (uaddr, UNKNOWN))
        free ((char *) uaddr);
    return (FALSE);
}

static void
brdcst (argc, argv)
    int argc;
    char **argv;
{
    enum clnt_stat rpc_stat;
    u_long prognum, vers;

    if (argc != 2)
    {
        usage ();
        exit (1);
    }
    prognum = getprognum (argv[0]);
    vers = getvers (argv[1]);
    rpc_stat = rpc_broadcast (prognum, vers, NULLPROC,
        (xdrproc_t) xdr_void, (char *) NULL,
        (xdrproc_t) xdr_void, (char *) NULL,
        (resultproc_t) reply_proc, NULL);
    if ((rpc_stat != RPC_SUCCESS) && (rpc_stat != RPC_TIMEDOUT))
    {
        fprintf (stderr, "rpcinfo: broadcast failed: %s\n",
            clnt_sperrno (rpc_stat));
        exit (1);
    }
    exit (0);
}

static bool_t

```

```
add_version (rs, vers)
{
    struct rpcbdump_short *rs;
    u_long vers;

    struct verslist *vl;

    for (vl = rs->vlist; vl; vl = vl->next)
        if (vl->vers == vers)
            break;
    if (vl)
        return (TRUE);
    vl = (struct verslist *) malloc (sizeof (struct verslist));
    if (vl == NULL)
        return (FALSE);
    vl->vers = vers;
    vl->next = rs->vlist;
    rs->vlist = vl;
    return (TRUE);
}

static bool_t
add_netid (rs, netid)
{
    struct rpcbdump_short *rs;
    char *netid;

    struct netidlist *nl;

    for (nl = rs->nlist; nl; nl = nl->next)
        if (strcmp (nl->netid, netid) == 0)
            break;
    if (nl)
        return (TRUE);
    nl = (struct netidlist *) malloc (sizeof (struct netidlist));
    if (nl == NULL)
        return (FALSE);
    nl->netid = netid;
    nl->next = rs->nlist;
    rs->nlist = nl;
    return (TRUE);
}

static void
rpcbdump (dumptype, netid, argc, argv)
{
    int dumptype;
    char *netid;
    int argc;
    char **argv;

    {
        rpcblist_ptr head = NULL;
        struct timeval minutetimeout;
        register CLIENT *client;
        struct rpcent *rpc;
        char *host;
        struct netidlist *nl;
    }
}
```

```

struct verslist *vl;
struct rpcbdump_short *rs, *rs_tail = NULL;
char buf[256];
enum clnt_stat clnt_st;
struct rpc_err err;
struct rpcbdump_short *rs_head = NULL;

if (argc > 1)
{
    usage ();
    exit (1);
}
if (argc == 1)
{
    host = argv[0];
    if (netid == NULL)
    {
        client = clnt_rpcbind_create (host, RPCBVERS, NULL);
    }
    else
    {
        struct netconfig *nconf;

        nconf = getnetconfig (netid);
        if (nconf == NULL)
        {
            nc_perror ("rpcinfo: invalid transport");
            exit (1);
        }
        client = getclnthandle (host, nconf, RPCBVERS, NULL);
        if (nconf)
            (void) freenetconfig (nconf);
    }
}
else
    client = local_rpcb (PMAPPROG, RPCBVERS);

if (client == (CLIENT *) NULL)
{
    clnt_pcreateerror ("rpcinfo: can't contact rpcbind");
    exit (1);
}
minutetimeout.tv_sec = 60;
minutetimeout.tv_usec = 0;
clnt_st = CLNT_CALL (client, RPCBPROC_DUMP, (xdrproc_t) xdr_void,
    NULL, (xdrproc_t) xdr_rpcblist_ptr, (char *) &head,
    minutetimeout);
if (clnt_st != RPC_SUCCESS)
{
    if ((clnt_st == RPC_PROGVERSMISMATCH) || (clnt_st == RPC_PROGUNAVAIL))
    {
        int vers;

        CLNT_GETERR (client, &err);
        if (err.re_vers.low == RPCBVERS4)

```

```

    {
        vers = RPCBVERS4;
        clnt_control (client, CLSET_VERS, (char *) &vers);
        clnt_st = CLNT_CALL (client, RPCBPROC_DUMP,
            (xdrproc_t) xdr_void, NULL,
            (xdrproc_t) xdr_rpcblist_ptr,
            (char *) &head, minutetimeout);
        if (clnt_st != RPC_SUCCESS)
            goto failed;
    }
    else
    {
        if (err.re_vers.high == PMAPVERS)
        {
            int high, low;
            struct pmaplist *pmaphead = NULL;
            rpcblist_ptr list, prev = NULL;

            vers = PMAPVERS;
            clnt_control (client, CLSET_VERS, (char *) &vers);
            clnt_st = CLNT_CALL (client, PMAPPROC_DUMP,
                (xdrproc_t) xdr_void, NULL,
                (xdrproc_t) xdr_pmaplist_ptr,
                (char *) &pmaphead, minutetimeout);
            if (clnt_st != RPC_SUCCESS)
                goto failed;
            /*
             * convert to rpcblist_ptr format
             */
            for (head = NULL; pmaphead != NULL;
                pmaphead = pmaphead->pml_next)
            {
                list = (rpcblist *) malloc (sizeof (rpcblist));
                if (list == NULL)
                    goto error;
                if (head == NULL)
                    head = list;
                else
                    prev->rpcb_next = (rpcblist_ptr) list;

                list->rpcb_next = NULL;
                list->rpcb_map.r_prog = pmaphead->pml_map.pm_prog;
                list->rpcb_map.r_vers = pmaphead->pml_map.pm_vers;
                if (pmaphead->pml_map.pm_prot == IPPROTO_UDP)
                    list->rpcb_map.r_netid = "udp";
                else if (pmaphead->pml_map.pm_prot == IPPROTO_TCP)
                    list->rpcb_map.r_netid = "tcp";
                else
                {
                    #define MAXLONG_AS_STRING "2147483648"
                    list->rpcb_map.r_netid =
                        malloc (strlen (MAXLONG_AS_STRING) + 1);
                    if (list->rpcb_map.r_netid == NULL)
                        goto error;

```

```

    sprintf (list->rpcb_map.r_netid, "%6ld",
    pmaphead->pml_map.pm_prot);
}
    list->rpcb_map.r_owner = UNKNOWN;
    low = pmaphead->pml_map.pm_port & 0xff;
    high = (pmaphead->pml_map.pm_port >> 8) & 0xff;
    list->rpcb_map.r_addr = strdup ("0.0.0.0.XXX.XXX");
    sprintf (&list->rpcb_map.r_addr[8], "%d.%d", high, low);
    prev = list;
}
}
}
}
    else
    { /* any other error */
failed:
    clnt_perror (client, "rpcinfo: can't contact rpcbind: ");
    exit (1);
}
    }
    if (head == NULL)
    {
        printf ("No remote programs registered.\n");
    }
    else if (dumptype == RPCBDUMP)
    {
        printf
        (" program version netid  address      service  owner\n");
        for (; head != NULL; head = head->rpcb_next)
        {
            printf ("%10u%5u  ",
            head->rpcb_map.r_prog, head->rpcb_map.r_vers);
            printf ("%9s ", head->rpcb_map.r_netid);
            printf ("%22s", head->rpcb_map.r_addr);
            rpc = getrpcbyname (head->rpcb_map.r_prog);
            if (rpc)
                printf (" %-10s", rpc->r_name);
            else
                printf (" %-10s", "-");
            printf (" %s\n", head->rpcb_map.r_owner);
        }
    }
    else if (dumptype == RPCBDUMP_SHORT)
    {
        for (; head != NULL; head = head->rpcb_next)
        {
            for (rs = rs_head; rs; rs = rs->next)
            if (head->rpcb_map.r_prog == rs->prog)
                break;
            if (rs == NULL)
            {
                rs = (struct rpcbdump_short *)
                malloc (sizeof (struct rpcbdump_short));
                if (rs == NULL)
                    goto error;
            }
        }
    }

```

```

        rs->next = NULL;
        if (rs_head == NULL)
        {
            rs_head = rs;
            rs_tail = rs;
        }
        else
        {
            rs_tail->next = rs;
            rs_tail = rs;
        }
        rs->prog = head->rpcb_map.r_prog;
        rs->owner = head->rpcb_map.r_owner;
        rs->nlist = NULL;
        rs->vlist = NULL;
    }
    if (add_version (rs, head->rpcb_map.r_vers) == FALSE)
        goto error;
    if (add_netid (rs, head->rpcb_map.r_netid) == FALSE)
        goto error;
}
    printf
("  program version(s) netid(s)          service  owner\n");
    for (rs = rs_head; rs; rs = rs->next)
    {
        char *p = buf;

        printf ("%10ld ", rs->prog);
        for (vl = rs->vlist; vl; vl = vl->next)
        {
            sprintf (p, "%d", vl->vers);
            p = p + strlen (p);
            if (vl->next)
                sprintf (p++, ",");
        }
        printf ("%10s", buf);
        buf[0] = '\0';
        for (nl = rs->nlist; nl; nl = nl->next)
        {
            strcat (buf, nl->netid);
            if (nl->next)
                sprintf (buf, ",");
        }
        printf ("%32s", buf);
        rpc = getrpcbynumber (rs->prog);
        if (rpc)
            printf (" %-11s", rpc->r_name);
        else
            printf (" %-11s", "-");
        printf (" %s\n", rs->owner);
    }
}
    clnt_destroy (client);
    return;

```

```

error:fprintf(stderr, "rpcinfo: no memory\n");
return;
}

static char nullstring[] = "\000";

static void
rpcbaddrlist (netid, argc, argv)
    char *netid;
    int argc;
    char **argv;
{
    rpcb_entry_list_ptr head = NULL;
    struct timeval minutetimeout;
    register CLIENT *client;
    struct rpcent *rpc;
    char *host;
    RPCB parms;
    struct netbuf *targaddr;

    if (argc != 3)
    {
        usage ();
        exit (1);
    }
    host = argv[0];
    if (netid == NULL)
    {
        client = clnt_rpcbind_create (host, RPCBVERS4, &targaddr);
    }
    else
    {
        struct netconfig *nconf;

        nconf = getnetconfig (netid);
        if (nconf == NULL)
        {
            nc_perror ("rpcinfo: invalid transport");
            exit (1);
        }
        client = getclnthandle (host, nconf, RPCBVERS4, &targaddr);
        if (nconf)
            (void) freenetconfig (nconf);
    }
    if (client == (CLIENT *) NULL)
    {
        clnt_pcreateerror ("rpcinfo: can't contact rpcbind");
        exit (1);
    }
    minutetimeout.tv_sec = 60;
    minutetimeout.tv_usec = 0;

    parms.r_prog = getprognum (argv[1]);
    parms.r_vers = getvers (argv[2]);
    parms.r_netid = client->cl_netid;

```



```

if (targaddr == NULL)
{
    parms.r_addr = nullstring; /* for XDRing */
}
else
{
    /*
     * We also send the remote system the address we
     * used to contact it in case it can help it
     * connect back with us
     */
    struct netconfig *nconf;

    nconf = getnetconfigent (client->cl_netid);
    if (nconf != NULL)
    {
        parms.r_addr = taddr2uaddr (nconf, targaddr);
        if (parms.r_addr == NULL)
            parms.r_addr = nullstring;
        freenetconfigent (nconf);
    }
    else
    {
        parms.r_addr = nullstring; /* for XDRing */
    }
    free (targaddr->buf);
    free (targaddr);
}
parms.r_owner = nullstring;

if (CLNT_CALL (client, RPCBPROC_GETADDRLIST, (xdrproc_t) xdr_rpcb,
(char *) &parms, (xdrproc_t) xdr_rpcb_entry_list_ptr,
(char *) &head, minutetimeout) != RPC_SUCCESS)
{
    clnt_perror (client, "rpcinfo: can't contact rpcbind: ");
    exit (1);
}
if (head == NULL)
{
    printf ("No remote programs registered.\n");
}
else
{
    printf
(" program vers tp_family/name/class address\t\t service\n");
    for (; head != NULL; head = head->rpcb_entry_next)
    {
        rpcb_entry *re;
        char buf[128];

        re = &head->rpcb_entry_map;
        printf ("%10u%3u ", parms.r_prog, parms.r_vers);
        sprintf (buf, "%s/%s/%s ",
            re->r_nc_protofmly, re->r_nc_proto,

```

```

    re->r_nc_semantics == NC_TPI_CLTS ? "clts" :
    re->r_nc_semantics == NC_TPI_COTS ? "cots" : "cots_ord");
    printf ("%24s", buf);
    printf ("%24s", re->r_maddr);
    rpc = getrpcbynumber (parms.r_prog);
    if (rpc)
        printf (" %-13s", rpc->r_name);
    else
        printf (" %-13s", "-");
    printf ("\n");
}
}
clnt_destroy (client);
return;
}

/*
 * monitor rpcbind
 */
static void
rpcbgetstat (argc, argv)
    int argc;
    char **argv;
{
    rpcb_stat_byvers inf;
    struct timeval minutetimeout;
    register CLIENT *client;
    char *host;
    int i, j;
    rpcbs_addrlist *pa;
    rpcbs_rmtcalllist *pr;
    int cnt, flen;
#define MAXFIELD64
    char fieldbuf[MAXFIELD];
#define MAXLINE256
    char linebuf[MAXLINE];
    char *cp, *lp;
    char *pmaphdr[] = {
        "NULL", "SET", "UNSET", "GETPORT",
        "DUMP", "CALLIT"
    };
    char *rpcb3hdr[] = {
        "NULL", "SET", "UNSET", "GETADDR", "DUMP", "CALLIT", "TIME",
        "U2T", "T2U"
    };
    char *rpcb4hdr[] = {
        "NULL", "SET", "UNSET", "GETADDR", "DUMP", "CALLIT", "TIME",
        "U2T", "T2U", "VERADDR", "INDRECT", "GETLIST", "GETSTAT"
    };
#define TABSTOP8

    if (argc >= 1)
    {
        host = argv[0];

```

```

    client = clnt_rpcbind_create (host, RPCBVERS4, NULL);
}
else
    client = local_rpcb (PMAPPROG, RPCBVERS4);
if (client == (CLIENT *) NULL)
{
    clnt_pcreateerror ("rpcinfo: can't contact rpcbind");
    exit (1);
}
minutetimeout.tv_sec = 60;
minutetimeout.tv_usec = 0;
memset ((char *) &inf, 0, sizeof (rpcb_stat_byvers));
if (CLNT_CALL (client, RPCBPROC_GETSTAT, (xdrproc_t) xdr_void, NULL,
(xdrproc_t) xdr_rpcb_stat_byvers, (char *) &inf,
minutetimeout) != RPC_SUCCESS)
{
    clnt_perror (client, "rpcinfo: can't contact rpcbind: ");
    exit (1);
}
printf ("PORTMAP (version 2) statistics\n");
lp = linebuf;
for (i = 0; i <= rpcb_highproc_2; i++)
{
    fieldbuf[0] = '\0';
    switch (i)
    {
case PMAPPROC_SET:
    sprintf (fieldbuf, "%d/", inf[RPCBVERS_2_STAT].setinfo);
    break;
case PMAPPROC_UNSET:
    sprintf (fieldbuf, "%d/", inf[RPCBVERS_2_STAT].unsetinfo);
    break;
case PMAPPROC_GETPORT:
    cnt = 0;
    for (pa = inf[RPCBVERS_2_STAT].addrinfo; pa; pa = pa->next)
        cnt += pa->success;
    sprintf (fieldbuf, "%d/", cnt);
    break;
case PMAPPROC_CALLIT:
    cnt = 0;
    for (pr = inf[RPCBVERS_2_STAT].rmtinfo; pr; pr = pr->next)
        cnt += pr->success;
    sprintf (fieldbuf, "%d/", cnt);
    break;
default:
    break; /* For the remaining ones */
    }
    cp = &fieldbuf[0] + strlen (fieldbuf);
    sprintf (cp, "%d", inf[RPCBVERS_2_STAT].info[i]);
    flen = strlen (fieldbuf);
    printf ("%s%s", pmaphdr[i],
spaces ((TABSTOP * (1 + flen / TABSTOP))
- strlen (pmaphdr[i])));
    sprintf (lp, "%s%s", fieldbuf,

```

```

        spaces (cnt = ((TABSTOP * (1 + flen / TABSTOP)) - flen));
        lp += (flen + cnt);
    }
    printf ("\n%s\n", linebuf);

    if (inf[RPCBVERS_2_STAT].info[PMAPPROC_CALLIT])
    {
        printf ("PMAP_RMTCALL call statistics\n");
        print_rmtcallstat (RPCBVERS_2_STAT, &inf[RPCBVERS_2_STAT]);
        printf ("\n");
    }

    if (inf[RPCBVERS_2_STAT].info[PMAPPROC_GETPORT])
    {
        printf ("PMAP_GETPORT call statistics\n");
        print_getaddrstat (RPCBVERS_2_STAT, &inf[RPCBVERS_2_STAT]);
        printf ("\n");
    }

    printf ("RPCBIND (version 3) statistics\n");
    lp = linebuf;
    for (i = 0; i <= rpcb_highproc_3; i++)
    {
        fieldbuf[0] = '\0';
        switch (i)
        {
        case RPCBPROC_SET:
            sprintf (fieldbuf, "%d/", inf[RPCBVERS_3_STAT].setinfo);
            break;
        case RPCBPROC_UNSET:
            sprintf (fieldbuf, "%d/", inf[RPCBVERS_3_STAT].unsetinfo);
            break;
        case RPCBPROC_GETADDR:
            cnt = 0;
            for (pa = inf[RPCBVERS_3_STAT].addrinfo; pa; pa = pa->next)
                cnt += pa->success;
            sprintf (fieldbuf, "%d/", cnt);
            break;
        case RPCBPROC_CALLIT:
            cnt = 0;
            for (pr = inf[RPCBVERS_3_STAT].rmtinfo; pr; pr = pr->next)
                cnt += pr->success;
            sprintf (fieldbuf, "%d/", cnt);
            break;
        default:
            break; /* For the remaining ones */
        }

        cp = &fieldbuf[0] + strlen (fieldbuf);
        sprintf (cp, "%d", inf[RPCBVERS_3_STAT].info[i]);
        flen = strlen (fieldbuf);
        printf ("%s%s", rpcb3hdr[i],
            spaces ((TABSTOP * (1 + flen / TABSTOP))
                - strlen (rpcb3hdr[i])));
        sprintf (lp, "%s%s", fieldbuf,
            spaces (cnt = ((TABSTOP * (1 + flen / TABSTOP)) - flen)));
    }

```

```

    lp += (flen + cnt);
}
printf ("\n%s\n\n", linebuf);

if (inf[RPCBVERS_3_STAT].info[RPCBPROC_CALLIT])
{
    printf ("RPCB_RMTCALL (version 3) call statistics\n");
    print_rmtcallstat (RPCBVERS_3_STAT, &inf[RPCBVERS_3_STAT]);
    printf ("\n");
}

if (inf[RPCBVERS_3_STAT].info[RPCBPROC_GETADDR])
{
    printf ("RPCB_GETADDR (version 3) call statistics\n");
    print_getaddrstat (RPCBVERS_3_STAT, &inf[RPCBVERS_3_STAT]);
    printf ("\n");
}

printf ("RPCBIND (version 4) statistics\n");

for (j = 0; j <= 9; j += 9)
    /* Just two iterations for printing */
    lp = linebuf;
    for (i = j; i <= max (8, rpcb_highproc_4 - 9 + j); i++)
    {
        fieldbuf[0] = '\0';
        switch (i)
        {
            case RPCBPROC_SET:
                sprintf (fieldbuf, "%d/", inf[RPCBVERS_4_STAT].setinfo);
                break;
            case RPCBPROC_UNSET:
                sprintf (fieldbuf, "%d/", inf[RPCBVERS_4_STAT].unsetinfo);
                break;
            case RPCBPROC_GETADDR:
                cnt = 0;
                for (pa = inf[RPCBVERS_4_STAT].addrinfo; pa; pa = pa->next)
                    cnt += pa->success;
                sprintf (fieldbuf, "%d/", cnt);
                break;
            case RPCBPROC_CALLIT:
                cnt = 0;
                for (pr = inf[RPCBVERS_4_STAT].rmtinfo; pr; pr = pr->next)
                    cnt += pr->success;
                sprintf (fieldbuf, "%d/", cnt);
                break;
            default:
                break; /* For the remaining ones */
        }
        cp = &fieldbuf[0] + strlen (fieldbuf);
        /*
        * XXX: We also add RPCBPROC_GETADDRLIST queries to
        * RPCB_GETADDR because rpcbind includes the
        * RPCB_GETADDRLIST successes in RPCB_GETADDR.

```

```

    */
    if (i != RPCBPROC_GETADDR)
        sprintf (cp, "%d", inf[RPCBVERS_4_STAT].info[i]);
    else
        sprintf (cp, "%d", inf[RPCBVERS_4_STAT].info[i] +
            inf[RPCBVERS_4_STAT].info[RPCBPROC_GETADDRLIST]);
    flen = strlen (fieldbuf);
    printf ("%s%s", rpcb4hdr[i],
        spaces ((TABSTOP * (1 + flen / TABSTOP))
            - strlen (rpcb4hdr[i])));
    sprintf (lp, "%s%s", fieldbuf,
        spaces (cnt = ((TABSTOP * (1 + flen / TABSTOP)) - flen)));
    lp += (flen + cnt);
}
    printf ("\n%s\n", linebuf);
}

if (inf[RPCBVERS_4_STAT].info[RPCBPROC_CALLIT] ||
    inf[RPCBVERS_4_STAT].info[RPCBPROC_INDIRECT])
{
    printf ("\n");
    printf ("RPCB_RMTCALL (version 4) call statistics\n");
    print_rmtcallstat (RPCBVERS_4_STAT, &inf[RPCBVERS_4_STAT]);
}

if (inf[RPCBVERS_4_STAT].info[RPCBPROC_GETADDR])
{
    printf ("\n");
    printf ("RPCB_GETADDR (version 4) call statistics\n");
    print_getaddrstat (RPCBVERS_4_STAT, &inf[RPCBVERS_4_STAT]);
}
clnt_destroy (client);
}

/*
 * Delete registration for this (prog, vers, netid)
 */
static void
deletereg (netid, argc, argv)
    char *netid;
    int argc;
    char **argv;
{
    struct netconfig *nconf = NULL;

    if (argc != 2)
    {
        usage ();
        exit (1);
    }
    if (netid)
    {
        nconf = getnetconfigt (netid);
        if (nconf == NULL)
        {

```

```

    fprintf(stderr, "rpcinfo: netid %s not supported\n", netid);
    exit(1);
}
}
if ((rpcb_unset(getprognum(argv[0]), getvers(argv[1]), nconf) == 0)
    {
        fprintf(stderr,
            "rpcinfo: Could not delete registration for prog %s version %s\n",
            argv[0], argv[1]);
        exit(1);
    }
}

/*
 * Create and return a handle for the given nconf.
 * Exit if cannot create handle.
 */
static CLIENT *
clnt_addr_create(address, nconf, prog, vers)
    char *address;
    struct netconfig *nconf;
    u_long prog;
    u_long vers;
{
    CLIENT *client;
    static struct netbuf *nbuf;
    static int fd = RPC_ANYFD;

    if (fd == RPC_ANYFD)
    {
        if ((fd = __rpc_nconf2fd(nconf)) == -1)
        {
            rpc_createerr.cf_stat = RPC_TLIERROR;
            clnt_pcreateerror("rpcinfo");
            exit(1);
        }
        /* Convert the uaddr to taddr */
        nbuf = uaddr2taddr(nconf, address);
        if (nbuf == NULL)
        {
            errx(1, "rpcinfo: no address for client handle");
            exit(1);
        }
    }
    client = clnt_tli_create(fd, nconf, nbuf, prog, vers, 0, 0);
    if (client == (CLIENT *) NULL)
    {
        clnt_pcreateerror("rpcinfo");
        exit(1);
    }
    return (client);
}

/*

```

```

* If the version number is given, ping that (prog, vers); else try to find
* the version numbers supported for that prog and ping all the versions.
* Remote rpcbind is not contacted for this service. The requests are
* sent directly to the services themselves.
*/
static void
addrping (address, netid, argc, argv)
    char *address;
    char *netid;
    int argc;
    char **argv;
{
    CLIENT *client;
    struct timeval to;
    enum clnt_stat rpc_stat;
    u_int32_t prognum, versnum, minvers, maxvers;
    struct rpc_err rpcerr;
    int failure = 0;
    struct netconfig *nconf;
    int fd;

    if (argc < 1 || argc > 2 || (netid == NULL))
    {
        usage ();
        exit (1);
    }
    nconf = getnetconfigt (netid);
    if (nconf == (struct netconfig *) NULL)
    {
        fprintf (stderr, "rpcinfo: Could not find %s\n", netid);
        exit (1);
    }
    to.tv_sec = 10;
    to.tv_usec = 0;
    prognum = getprognum (argv[0]);
    if (argc == 1)
    {
        /* Version number not known */
        /*
         * A call to version 0 should fail with a program/version
         * mismatch, and give us the range of versions supported.
         */
        versnum = MIN_VERS;
    }
    else
    {
        versnum = getvers (argv[1]);
    }
    client = clnt_addr_create (address, nconf, prognum, versnum);
    rpc_stat = CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
(char *) NULL, (xdrproc_t) xdr_void,
(char *) NULL, to);
    if (argc == 2)
    {
        /* Version number was known */
        if (pstatus (client, prognum, versnum) < 0)

```



```

failure = 1;
(void) CLNT_DESTROY (client);
if (failure)
exit (1);
return;
}
/* Version number not known */
(void) CLNT_CONTROL (client, CLSET_FD_NCLOSE, (char *) NULL);
(void) CLNT_CONTROL (client, CLGET_FD, (char *) &fd);
if (rpc_stat == RPC_PROGVERSMISMATCH)
{
    clnt_geterr (client, &rpcerr);
    minvers = rpcerr.re_vers.low;
    maxvers = rpcerr.re_vers.high;
}
else if (rpc_stat == RPC_SUCCESS)
{
    /*
     * Oh dear, it DOES support version 0.
     * Let's try version MAX_VERS.
     */
    (void) CLNT_DESTROY (client);
    client = clnt_addr_create (address, nconf, prognum, MAX_VERS);
    rpc_stat = CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
        (char *) NULL, (xdrproc_t) xdr_void,
        (char *) NULL, to);
    if (rpc_stat == RPC_PROGVERSMISMATCH)
    {
        clnt_geterr (client, &rpcerr);
        minvers = rpcerr.re_vers.low;
        maxvers = rpcerr.re_vers.high;
    }
    else if (rpc_stat == RPC_SUCCESS)
    {
        /*
         * It also supports version MAX_VERS.
         * Looks like we have a wise guy.
         * OK, we give them information on all
         * 4 billion versions they support...
         */
        minvers = 0;
        maxvers = MAX_VERS;
    }
    else
    {
        (void) pstatus (client, prognum, MAX_VERS);
        exit (1);
    }
}
else
{
    (void) pstatus (client, prognum, (u_long) 0);
    exit (1);
}

```

```

(void) CLNT_DESTROY (client);
for (versnum = minvers; versnum <= maxvers; versnum++)
{
    client = clnt_addr_create (address, nconf, prognum, versnum);
    rpc_stat = CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
(char *) NULL, (xdrproc_t) xdr_void,
(char *) NULL, to);
    if (pstatus (client, prognum, versnum) < 0)
failure = 1;
    (void) CLNT_DESTROY (client);
}
(void) close (fd);
if (failure)
    exit (1);
return;
}

/*
 * If the version number is given, ping that (prog, vers); else try to find
 * the version numbers supported for that prog and ping all the versions.
 * Remote rpcbind is *contacted* for this service. The requests are
 * then sent directly to the services themselves.
 */
static void
progping (netid, argc, argv)
    char *netid;
    int argc;
    char **argv;
{
    CLIENT *client;
    struct timeval to;
    enum clnt_stat rpc_stat;
    u_int32_t prognum, versnum, minvers, maxvers;
    struct rpc_err rpcerr;
    int failure = 0;
    struct netconfig *nconf;

    if (argc < 2 || argc > 3 || (netid == NULL))
    {
        usage ();
        exit (1);
    }
    prognum = getprognum (argv[1]);
    if (argc == 2)
    {
        /* Version number not known */
        /*
         * A call to version 0 should fail with a program/version
         * mismatch, and give us the range of versions supported.
         */
        versnum = MIN_VERS;
    }
    else
    {
        {
            versnum = getvers (argv[2]);
        }
    }
}

```

```

    if (netid)
    {
        nconf = getnetconfigent (netid);
        if (nconf == (struct netconfig *) NULL)
        {
            fprintf (stderr, "rpcinfo: Could not find %s\n", netid);
            exit (1);
        }
        client = clnt_tp_create (argv[0], prognum, versnum, nconf);
    }
    else
    {
        client = clnt_create (argv[0], prognum, versnum, "NETPATH");
    }
    if (client == (CLIENT *) NULL)
    {
        clnt_pcreateerror ("rpcinfo");
        exit (1);
    }
    to.tv_sec = 10;
    to.tv_usec = 0;
    rpc_stat = CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
(char *) NULL, (xdrproc_t) xdr_void,
(char *) NULL, to);
    if (argc == 3)
    {
        /* Version number was known */
        if (pstatus (client, prognum, versnum) < 0)
        failure = 1;
        (void) CLNT_DESTROY (client);
        if (failure)
        exit (1);
        return;
    }
    /* Version number not known */
    if (rpc_stat == RPC_PROGVERSMISMATCH)
    {
        clnt_geterr (client, &rpcerr);
        minvers = rpcerr.re_vers.low;
        maxvers = rpcerr.re_vers.high;
    }
    else if (rpc_stat == RPC_SUCCESS)
    {
        /*
         * Oh dear, it DOES support version 0.
         * Let's try version MAX_VERS.
         */
        versnum = MAX_VERS;
        (void) CLNT_CONTROL (client, CLSET_VERS, (char *) &versnum);
        rpc_stat = CLNT_CALL (client, NULLPROC,
(xdrproc_t) xdr_void, (char *) NULL,
(xdrproc_t) xdr_void, (char *) NULL, to);
        if (rpc_stat == RPC_PROGVERSMISMATCH)
        {

```

```

    clnt_geterr (client, &rpcerr);
    minvers = rpcerr.re_vers.low;
    maxvers = rpcerr.re_vers.high;
}
    else if (rpc_stat == RPC_SUCCESS)
{
    /*
     * It also supports version MAX_VERS.
     * Looks like we have a wise guy.
     * OK, we give them information on all
     * 4 billion versions they support...
     */
    minvers = 0;
    maxvers = MAX_VERS;
}
    else
    {
        (void) pstatus (client, prognum, MAX_VERS);
        exit (1);
    }
}
else
{
    (void) pstatus (client, prognum, (u_long) 0);
    exit (1);
}
}
for (versnum = minvers; versnum <= maxvers; versnum++)
{
    (void) CLNT_CONTROL (client, CLSET_VERS, (char *) &versnum);
    rpc_stat = CLNT_CALL (client, NULLPROC, (xdrproc_t) xdr_void,
        (char *) NULL, (xdrproc_t) xdr_void,
        (char *) NULL, to);
    if (pstatus (client, prognum, versnum) < 0)
        failure = 1;
}
(void) CLNT_DESTROY (client);
if (failure)
    exit (1);
return;
}

static void
usage ()
{
    fprintf (stderr, "Usage: rpcinfo [-m | -s] [host]\n");
#ifdef PORTMAP
    fprintf (stderr, "    rpcinfo -p [host]\n");
#endif
    fprintf (stderr, "    rpcinfo -T netid host prognum [versnum]\n");
    fprintf (stderr, "    rpcinfo -l host prognum versnum\n");
#ifdef PORTMAP
    fprintf (stderr,
        "    rpcinfo [-n portnum] -u | -t host prognum [versnum]\n");
#endif
    fprintf (stderr,

```

```
"    rpcinfo -a serv_address -T netid prognum [version]\n");
fprintf(stderr, "    rpcinfo -b prognum versnum\n");
fprintf(stderr, "    rpcinfo -d [-T netid] prognum versnum\n");
}

static u_long
getprognum (arg)
    char *arg;
{
    char *strptr;
    register struct rpcent *rpc;
    register u_long prognum;
    char *tpr = arg;

    while (*tpr && isdigit (*tpr++));
    if (*tpr || isalpha (*(tpr - 1)))
    {
        rpc = getrpcbyname (arg);
        if (rpc == NULL)
        {
            fprintf(stderr, "rpcinfo: %s is unknown service\n", arg);
            exit (1);
        }
        prognum = rpc->r_number;
    }
    else
    {
        prognum = strtol (arg, &strptr, 10);
        if (strptr == arg || *strptr != '\0')
        {
            fprintf(stderr, "rpcinfo: %s is illegal program number\n", arg);
            exit (1);
        }
    }
    return (prognum);
}

static u_long
getvers (arg)
    char *arg;
{
    char *strptr;
    register u_long vers;

    vers = (int) strtol (arg, &strptr, 10);
    if (strptr == arg || *strptr != '\0')
    {
        fprintf(stderr, "rpcinfo: %s is illegal version number\n", arg);
        exit (1);
    }
    return (vers);
}

/*
```

```

* This routine should take a pointer to an "rpc_err" structure, rather than
* a pointer to a CLIENT structure, but "clnt_perror" takes a pointer to
* a CLIENT structure rather than a pointer to an "rpc_err" structure.
* As such, we have to keep the CLIENT structure around in order to print
* a good error message.
*/
static int
pstatus (client, prog, vers)
    register CLIENT *client;
    u_long prog;
    u_long vers;
{
    struct rpc_err rpcerr;

    clnt_geterr (client, &rpcerr);
    if (rpcerr.re_status != RPC_SUCCESS)
    {
        clnt_perror (client, "rpcinfo");
        printf ("program %lu version %lu is not available\n", prog, vers);
        return (-1);
    }
    else
    {
        printf ("program %lu version %lu ready and waiting\n", prog, vers);
        return (0);
    }
}

static CLIENT *
clnt_rpcbind_create (host, rpcbversnum, targaddr)
    char *host;
    int rpcbversnum;
    struct netbuf **targaddr;
{
    static char *tlist[3] = {
        "circuit_n", "circuit_v", "datagram_v"
    };
    int i;
    struct netconfig *nconf;
    CLIENT *clnt = NULL;
    void *handle;

    rpc_createerr.cf_stat = RPC_SUCCESS;
    for (i = 0; i < 3; i++)
    {
        if ((handle = __rpc_setconf (tlist[i])) == NULL)
            continue;
        while (clnt == (CLIENT *) NULL)
        {
            if ((nconf = __rpc_getconf (handle)) == NULL)
            {
                if (rpc_createerr.cf_stat == RPC_SUCCESS)
                    rpc_createerr.cf_stat = RPC_UNKNOWNPROTO;
                break;
            }

```

```

    clnt = getclnthandle (host, nconf, rpcbversnum, targaddr);
}
    if (clnt)
break;
    __rpc_endconf (handle);
}
    return (clnt);
}

static CLIENT *
getclnthandle (host, nconf, rpcbversnum, targaddr)
    char *host;
    struct netconfig *nconf;
    u_long rpcbversnum;
    struct netbuf **targaddr;
{
    struct netbuf addr;
    struct addrinfo hints, *res;
    CLIENT *client = NULL;

    /* Get the address of the rpcbind */
    memset (&hints, 0, sizeof hints);
    if ((getaddrinfo (host, "rpcbind", &hints, &res) != 0) &&
        (getaddrinfo (host, "portmapper",&hints, &res) != 0))
    {
        rpc_createerr.cf_stat = RPC_N2AXLATEFAILURE;
        return (NULL);
    }
    addr.len = addr.maxlen = res->ai_addrlen;
    addr.buf = res->ai_addr;
    client = clnt_tli_create (RPC_ANYFD, nconf, &addr, RPCBPROG,
        rpcbversnum, 0, 0);
    if (client)
    {
        if (targaddr != NULL)
        {
            *targaddr = (struct netbuf *) malloc (sizeof (struct netbuf));
            if (*targaddr != NULL)
            {
                (*targaddr)->maxlen = addr.maxlen;
                (*targaddr)->len = addr.len;
                (*targaddr)->buf = (char *) malloc (addr.len);
                if ((*targaddr)->buf != NULL)
                {
                    memcpy ((*targaddr)->buf, addr.buf, addr.len);
                }
            }
        }
    }
    else
    {
        if (rpc_createerr.cf_stat == RPC_TLIERROR)
        {
            /*

```

```
* Assume that the other system is dead; this is a
* better error to display to the user.
*/
rpc_createerr.cf_stat = RPC_RPCBFAILURE;
rpc_createerr.cf_error.re_status = RPC_FAILED;
}
}
freeaddrinfo (res);
return (client);
}

static void
print_rmtcallstat (rtype, info)
    int rtype;
    rpcb_stat *info;
{
    register rpcbs_rmtcalllist_ptr pr;
    struct rpcent *rpc;

    if (rtype == RPCBVERS_4_STAT)
        printf ("prog\ttvers\tproc\tnetid\tindirect success failure\n");
    else
        printf ("prog\ttvers\tproc\tnetid\tsuccess\tfailure\n");
    for (pr = info->rmtinfo; pr; pr = pr->next)
    {
        rpc = getrpcbynumber (pr->prog);
        if (rpc)
            printf ("% -16s", rpc->r_name);
        else
            printf ("% -16d", pr->prog);
        printf ("%d\t%d\t%s\t", pr->vers, pr->proc, pr->netid);
        if (rtype == RPCBVERS_4_STAT)
            printf ("%d\t", pr->indirect);
        printf ("%d\t%d\n", pr->success, pr->failure);
    }
}

static void
print_getaddrstat (rtype, info)
    int rtype;
    rpcb_stat *info;
{
    rpcbs_addrlist_ptr al;
    register struct rpcent *rpc;

    printf ("prog\ttvers\tnetid\t success\tfailure\n");
    for (al = info->addrinfo; al; al = al->next)
    {
        rpc = getrpcbynumber (al->prog);
        if (rpc)
            printf ("% -16s", rpc->r_name);
        else
            printf ("% -16d", al->prog);
        printf ("%d\t%s\t % -12d\t%d\n",
            al->vers, al->netid, al->success, al->failure);
    }
}
```



```
    }  
}  
  
static char *  
spaces (howmany)  
    int howmany;  
{  
    static char space_array[] =/* 64 spaces */  
        "                                     ";  
  
    if (howmany <= 0 || howmany > sizeof (space_array))  
    {  
        return ("");  
    }  
    return (&space_array[sizeof (space_array) - howmany - 1]);  
}
```

1.119 rsync

1.119.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.120 rsyslog

1.120.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form,

including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute,

alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are

covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

1.121 run-postinsts

1.121.1 Available under license

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Different components of OpenEmbedded are under different licenses (a mix of MIT and GPLv2). Please see:

- meta/COPYING.GPLv2 (GPLv2)
- meta/COPYING.MIT (MIT)
- meta-selftest/COPYING.MIT (MIT)
- meta-skeleton/COPYING.MIT (MIT)

All metadata is MIT licensed unless otherwise stated. Source code included in tree for individual recipes is under the LICENSE stated in the associated recipe (.bb file) unless otherwise stated.

License information for any other files is either explicitly stated or defaults to GPL version 2.

1.122 screen

1.122.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <<http://www.gnu.org/licenses/>>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <<http://www.gnu.org/philosophy/why-not-lgpl.html>>.

/* Copyright (c) 2008, 2009

```
* Juergen Weigert (jnweiger@immd4.informatik.uni-erlangen.de)
* Michael Schroeder (mlschroe@immd4.informatik.uni-erlangen.de)
* Micah Cowan (micah@cowan.name)
* Sadrul Habib Chowdhury (sadrul@users.sourceforge.net)
```

* Copyright (c) 1993-2002, 2003, 2005, 2006, 2007

```
* Juergen Weigert (jnweiger@immd4.informatik.uni-erlangen.de)
* Michael Schroeder (mlschroe@immd4.informatik.uni-erlangen.de)
```

* Copyright (c) 1987 Oliver Laumann

*

```
* This program is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation; either version 3, or (at your option)
* any later version.
```

*

```
* This program is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
```

*

```
* You should have received a copy of the GNU General Public License
* along with this program (see the file COPYING); if not, see
* http://www.gnu.org/licenses/, or contact Free Software Foundation, Inc.,
* 51 Franklin Street, Fifth Floor, Boston, MA 02111-1301 USA
```

*

```
*****
```

```
* $Id$ GNU
```

```
*/
```

```
#include "os.h"

#if defined(__STDC__)
# ifndef __P
#  define __P(a) a
# endif
#else
# ifndef __P
#  define __P(a) ()
# endif
# define const
#endif

#include "osdef.h"

#include "ansi.h"
#include "sched.h"
#include "acls.h"
#include "comm.h"
#include "layer.h"
#include "term.h"

#ifdef DEBUG
# define STATIC/* a function that the debugger should see */
#else
# define STATIC static
#endif

#ifdef DEBUG
# define DEBUGDIR "/tmp/debug"
# define debugf(a)    do {if(dfp){fprintf a;fflush(dfp);}} while (0)
# define debug(x)     debugf((dfp,x))
# define debug1(x,a)  debugf((dfp,x,a))
# define debug2(x,a,b) debugf((dfp,x,a,b))
# define debug3(x,a,b,c) debugf((dfp,x,a,b,c))
extern FILE *dfp;
#else
# define debugf(a)    do {} while (0)
# define debug(x)     debugf(x)
# define debug1(x,a)  debugf(x)
# define debug2(x,a,b) debugf(x)
# define debug3(x,a,b,c) debugf(x)
#endif

#ifdef DEBUG
# define NOASSERT
#endif

#ifdef NOASSERT
# if defined(__STDC__)
#  define ASSERT(lousy_cpp) do {if (!(lousy_cpp)) {if (!dfp) opendebg(0, 1);debug2("ASSERT("#lousy_cpp")
failed file %s line %d\n", __FILE__, __LINE__);abort();}} while (0)

```

```

# else
#  define ASSERT(lousy_cpp) do {if (!(lousy_cpp)) {if (!dfp) opendebug(0, 1);debug2("ASSERT(lousy_cpp)
failed file %s line %d\n", __FILE__, __LINE__);abort();}} while (0)
# endif
#else
# define ASSERT(lousy_cpp) do {} while (0)
#endif

/* here comes my own Free: jw. */
#define Free(a) {if ((a) == 0) abort(); else free((void *) (a)); (a)=0;}

#define Ctrl(c) ((c)&037)

#define MAXSTR768
#define MAXARGS 64
#define MSGWAIT 5
#define MSGMINWAIT 1
#define SILENCEWAIT30

/*
 * if a nasty user really wants to try a history of 3000 lines on all 10
 * windows, he will allocate 8 MegaBytes of memory, which is quite enough.
 */
#define MAXHISTHEIGHT3000
#define DEFAULTHISTHEIGHT100
#if defined(NAME_MAX) && NAME_MAX < 16
# define DEFAULT_BUFFERFILE "/tmp/screen-xchg"
#else
# define DEFAULT_BUFFERFILE "/tmp/screen-exchange"
#endif

#if defined(hpux) && !(defined(VSUSP) && defined(VDSUSP) && defined(VWERASE) &&
defined(VLNEXT))
# define HPUX_LTCHARS_HACK
#endif

struct mode
{
#ifdef POSIX
    struct termios tio;
# ifdef HPUX_LTCHARS_HACK
    struct ltchars m_ltchars;
# endif /* HPUX_LTCHARS_HACK */
#else /* POSIX */
# ifdef TERMIO
    struct termio tio;
# ifdef CYTERMIO
    int m_mapkey;
    int m_mapscreen;
    int m_backspace;
# endif
# else /* TERMIO */
    struct sgtyb m_ttyb;
    struct tchars m_tchars;

```



```

    struct ltchars m_ltchars;
    int m_ldisc;
    int m_lmode;
# endif /* TERMIO */
# endif /* POSIX */
# if defined(KANJI) && defined(TIOCKSET)
    struct jtchars m_jtchars;
    int m_knjmode;
# endif
};

/* #include "logfile.h" */ /* (requires stat.h) struct logfile */
#include "image.h"
#include "canvas.h"
#include "display.h"
#include "window.h"

/*
 * Parameters for the Detach() routine
 */
#define D_DETACH0
#define D_STOP1
#define D_REMOTE2
#define D_POWER 3
#define D_REMOTE_POWER4
#define D_LOCK5
#define D_HANGUP6

/*
 * Here are the messages the attacher sends to the backend
 */
#define MSG_CREATE0
#define MSG_ERROR1
#define MSG_ATTACH2
#define MSG_CONT3
#define MSG_DETACH4
#define MSG_POW_DETACH5
#define MSG_WINCH6
#define MSG_HANGUP7
#define MSG_COMMAND8
#define MSG_QUERY 9

/*
 * versions of struct msg:
 * 0:screen version 3.6.6 (version count introduced)
 * 1:screen version 4.1.0devel(revisions e3fc19a upto 8147d08)
 * A few revisions after 8147d08 incorrectly
 * carried version 1, but should have carried 2.
 * 2:screen version 4.1.0devel(revisions 8b46d8a upto YYYYYYYY)
 * 3:screen version 4.2.0(was incorrectly originally. Patched here)
 * 4:screen version 4.2.1(bumped once again due to changed terminal and login length)
 * 5: screen version 4.4.0(fix screenterm size)
 */

```

```

#define MSG_VERSION5

#define MSG_REVISION(('m'<<24) | ('s'<<16) | ('g'<<8) | MSG_VERSION)
struct msg
{
    int protocol_revision; /* reduce harm done by incompatible messages */
    int type;
    char m_tty[MAXPATHLEN]; /* ttyname */
    union
    {
        struct
        {
            int lflag;
            int aflag;
            int flowflag;
            int hheight; /* size of scrollbar buffer */
            int nargs;
            char line[MAXPATHLEN];
            char dir[MAXPATHLEN];
            char screenterm[MAXTERMLEN + 1]; /* is screen really "screen" ? */
        }
        create;
        struct
        {
            char auser[MAXLOGINLEN + 1]; /* username */
            int apid; /* pid of frontend */
            int adaptflag; /* adapt window size? */
            int lines, columns; /* display size */
            char preselect[20];
            int esc; /* his new escape character unless -1 */
            int meta_esc; /* his new meta esc character unless -1 */
            char envterm[MAXTERMLEN + 1]; /* terminal type */
            int encoding; /* encoding of display */
            int detachfirst; /* whether to detach remote sessions first */
        }
        attach;
        struct
        {
            char duser[MAXLOGINLEN + 1]; /* username */
            int dpid; /* pid of frontend */
        }
        detach;
        struct
        {
            char auser[MAXLOGINLEN + 1]; /* username */
            int nargs;
            char cmd[MAXPATHLEN]; /* command */
            int apid; /* pid of frontend */
            char preselect[20];
            char writeback[MAXPATHLEN]; /* The socket to write the result.
            Only used for MSG_QUERY */
        }
        command;
        char message[MAXPATHLEN * 2];
    } m;
}

```

```

};

/*
 * And the signals the attacher receives from the backend
 */
#define SIG_BYESIGHUP
#define SIG_POWER_BYESIGUSR1
#define SIG_LOCKSIGUSR2
#define SIG_STOPSIGTSTP
#ifdef SIGIO
#define SIG_NODEBUGSIGIO /* triggered by command 'debug off' */
#endif

#define BELL(Ctrl('g'))
#define VBELLWAIT1 /* No. of seconds a vbell will be displayed */

#define BELL_ON0 /* No bell has occurred in the window */
#define BELL_FOUND1 /* A bell has occurred, but user not yet notified */
#define BELL_DONE2 /* A bell has occurred, user has been notified */

#define BELL_VISUAL3 /* A bell has occurred in fore win, notify him visually */

#define MON_OFF 0 /* Monitoring is off in the window */
#define MON_ON1 /* No activity has occurred in the window */
#define MON_FOUND2 /* Activity has occurred, but user not yet notified */
#define MON_DONE3 /* Activity has occurred, user has been notified */

#define DUMP_TERMCAP0 /* WriteFile() options */
#define DUMP_HARDCOPY1
#define DUMP_EXCHANGE2
#define DUMP_SCROLLBACK 3

#define SILENCE_OFF0 /* Not checking for silence */
#define SILENCE_ON1 /* Window being monitored for silence */
#define SILENCE_FOUND 2 /* Window is silent */
#define SILENCE_DONE 3 /* Window is silent and user is notified */

extern char strnomem[];

/*
 * line modes used by Input()
 */
#define INP_COOKED0
#define INP_NOECHO1
#define INP_RAW2
#define INP EVERY4

#ifdef MULTIUSER
struct acl
{
    struct acl *next;
    char *name;

```

```
};
#endif

/* register list */
#define MAX_PLOP_DEFS 256

struct baud_values
{
    int idx; /* the index in the bsd-is padding lookup table */
    int bps; /* bits per seconds */
    int sym; /* symbol defined in ttydev.h */
};

/*
 * windowlist orders
 */
#define WLIST_NUM 0
#define WLIST_MRU 1
#define WLIST_NESTED 2
```

1.123 sed

1.123.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as

changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no

more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as

part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in

reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For

purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may

otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY

APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```


You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

```
/* GNU SED, a batch stream editor.
Copyright (C) 1989,90,91,92,93,94,95,98,99,2002,2003
Free Software Foundation, Inc.
```

```
This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 3, or (at your option)
any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. */
```

```
#include "config.h"
#include "basicdefs.h"
#include "regex.h"
#include <stdio.h>
#include "unlocked-io.h"

#include "utils.h"
```

```

/* Struct vector is used to describe a compiled sed program. */
struct vector {
    struct sed_cmd *v; /* a dynamically allocated array */
    size_t v_allocated; /* ... number slots allocated */
    size_t v_length; /* ... number of slots in use */
};

/* This structure tracks files used by sed so that they may all be
   closed cleanly at normal program termination. A flag is kept that tells
   if a missing newline was encountered, so that it is added on the
   next line and the two lines are not concatenated. */
struct output {
    char *name;
    bool missing_newline;
    FILE *fp;
    struct output *link;
};

struct text_buf {
    char *text;
    size_t text_length;
};

struct regex {
    regex_t pattern;
    int flags;
    size_t sz;
    char re[1];
};

enum replacement_types {
    REPL_ASIS = 0,
    REPL_UPPERCASE = 1,
    REPL_LOWERCASE = 2,
    REPL_UPPERCASE_FIRST = 4,
    REPL_LOWERCASE_FIRST = 8,
    REPL_MODIFIERS = REPL_UPPERCASE_FIRST | REPL_LOWERCASE_FIRST,

    /* These are given to aid in debugging */
    REPL_UPPERCASE_UPPERCASE = REPL_UPPERCASE_FIRST | REPL_UPPERCASE,
    REPL_UPPERCASE_LOWERCASE = REPL_UPPERCASE_FIRST | REPL_LOWERCASE,
    REPL_LOWERCASE_UPPERCASE = REPL_LOWERCASE_FIRST | REPL_UPPERCASE,
    REPL_LOWERCASE_LOWERCASE = REPL_LOWERCASE_FIRST | REPL_LOWERCASE
};

enum text_types {
    TEXT_BUFFER,
    TEXT_REPLACEMENT,
    TEXT_REGEX
};

enum posixicity_types {
    POSIXLY_EXTENDED, /* with GNU extensions */
    POSIXLY_CORRECT, /* with POSIX-compatible GNU extensions */
};

```

```

    POSIXLY_BASIC/* pedantically POSIX */
};

enum addr_state {
    RANGE_INACTIVE/* never been active */
    RANGE_ACTIVE/* between first and second address */
    RANGE_CLOSED/* like RANGE_INACTIVE, but range has ended once */
};

enum addr_types {
    ADDR_IS_NULL/* null address */
    ADDR_IS_REGEX/* a.addr_regex is valid */
    ADDR_IS_NUM/* a.addr_number is valid */
    ADDR_IS_NUM_MOD/* a.addr_number is valid, addr_step is modulo */
    ADDR_IS_STEP/* address is +N (only valid for addr2) */
    ADDR_IS_STEP_MOD/* address is ~N (only valid for addr2) */
    ADDR_IS_LAST/* address is $ */
};

struct addr {
    enum addr_types addr_type;
    countT addr_number;
    countT addr_step;
    struct regex *addr_regex;
};

struct replacement {
    char *prefix;
    size_t prefix_length;
    int subst_id;
    enum replacement_types repl_type;
    struct replacement *next;
};

struct subst {
    struct regex *regex;
    struct replacement *replacement;
    countT numb;/* if >0, only substitute for match number "numb" */
    struct output *outf;/* 'w' option given */
    unsigned global : 1;/* 'g' option given */
    unsigned print : 2;/* 'p' option given (before/after eval) */
    unsigned eval : 1;/* 'e' option given */
    unsigned max_id : 4; /* maximum backreference on the RHS */
};

#ifdef REG_PERL
/* This is the structure we store register match data in. See
   regex.texinfo for a full description of what registers match. */
struct re_registers
{
    unsigned num_regs;
    regoff_t *start;
    regoff_t *end;
};

```

```

};
#endif

struct sed_cmd {
    struct addr *a1; /* save space: usually is NULL */
    struct addr *a2;

    /* See description the enum, above. */
    enum addr_state range_state;

    /* Non-zero if command is to be applied to non-matches. */
    char addr_bang;

    /* The actual command character. */
    char cmd;

    /* auxiliary data for various commands */
    union {
        /* This structure is used for a, i, and c commands. */
        struct text_buf cmd_txt;

        /* This is used for the l, q and Q commands. */
        int int_arg;

        /* This is used for the {}, b, and t commands. */
        countT jump_index;

        /* This is used for the r command. */
        char *fname;

        /* This is used for the hairy s command. */
        struct subst *cmd_subst;

        /* This is used for the w command. */
        struct output *outf;

        /* This is used for the R command. */
        FILE *fp;

        /* This is used for the y command. */
        unsigned char *translate;
        char **translatemb;
    } x;
};

void bad_prog (const char *why);
size_t normalize_text (char *text, size_t len, enum text_types buftype);
struct vector *compile_string (struct vector *, char *str, size_t len);
struct vector *compile_file (struct vector *, const char *cmdfile);
void check_final_program (struct vector *);
void rewind_read_files (void);

```

```
void finish_program (struct vector *);

struct regex *compile_regex (struct buffer *b, int flags, int needed_sub);
int match_regex (struct regex *regex,
char *buf, size_t buflen, size_t buf_start_offset,
struct re_registers *regarray, int regsize);
#ifdef DEBUG_LEAKS
void release_regex (struct regex *);
#endif

int process_files (struct vector *, char **argv);

int main (int, char **);

extern void fmt (const char *line, const char *line_end, int max_length, FILE *output_file);

extern int extended_regexp_flags;

/* one-byte buffer delimiter */
extern char buffer_delimiter;

/* If set, fflush(stdout) on every line output,
and turn off stream buffering on inputs. */
extern bool unbuffered;

/* If set, don't write out the line unless explicitly told to. */
extern bool no_default_output;

/* If set, reset line counts on every new file. */
extern bool separate_files;

/* If set, follow symlinks when invoked with -i option */
extern bool follow_symlinks;

/* Do we need to be pedantically POSIX compliant? */
extern enum posixicity_types posixicity;

/* How long should the 'l' command's output line be? */
extern countT lcmd_out_line_len;

/* How do we edit files in-place? (we don't if NULL) */
extern char *in_place_extension;

/* The mode to use to read and write files, either "rt"/"w" or "rb"/"wb". */
extern char *read_mode;
extern char *write_mode;

/* Should we use EREs? */
extern bool use_extended_syntax_p;

/* Declarations for multibyte character sets. */
extern int mb_cur_max;
extern bool is_utf8;
```

```
#define MBRTOWC(pwc, s, n, ps) \
    (mb_cur_max == 1 ? \
    (*(pwc) = btowc (*(unsigned char *) (s)), 1) : \
    mbrtowc ((pwc), (s), (n), (ps)))

#define WCRTOMB(s, wc, ps) \
    (mb_cur_max == 1 ? \
    (*(s) = wctob ((wint_t) (wc)), 1) : \
    wctomb ((s), (wc), (ps)))

#define MBSINIT(s) \
    (mb_cur_max == 1 ? 1 : mbsinit ((s)))

#define MBRLen(s, n, ps) \
    (mb_cur_max == 1 ? 1 : mbrtowc (NULL, s, n, ps))

#define BRLEN(ch, ps) \
    (mb_cur_max == 1 ? 1 : brlen (ch, ps))

extern int brlen (int ch, mbstate_t *ps);
extern void initialize_mbcs (void);
```

1.124 setserial

1.124.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted

interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision` (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
/*
 * version.h --- Defines the version number of setserial
 *
 * Copyright 1995, 1996, 1997, 1998 by Theodore Ts'o. This file may be
 * redistributed under the GNU Public License.
 */

#define SETSERIAL_VERSION "2.17"
#define SETSERIAL_DATE "27-Jan-2000"
```

1.125 shadow

1.125.1 Available under license

NOTE:

This license has been obsoleted by the change to the BSD-style copyright. You may continue to use this license if you wish, but you are under no obligation to do so.

(*

This document is freely plagiarised from the 'Artistic Licence', distributed as part of the Perl v4.0 kit by Larry Wall, which is available from most major archive sites. I stole it from CrackLib.

\$Id\$

*)

This documents purpose is to state the conditions under which this Package (See definition below) viz: "Shadow", the Shadow Password Suite which is held by Julianne Frances Haugh, may be copied, such that the copyright holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

So there.

Definitions:

A "Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification, or segments thereof.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when AND WHY you changed that file, and provided that you do at least ONE of the following:
 - a) place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
 - b) use the modified Package only within your corporation or organization.
 - c) rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide separate documentation for each non-standard executable that clearly documents how it differs from the Standard Version.
 - d) make other distribution arrangements with the Copyright Holder.
4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:
 - a) distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.
 - b) accompany the distribution with the machine-readable source of the Package with your modifications.
 - c) accompany any non-standard executables with their corresponding Standard Version executables, giving the non-standard executables non-standard names, and clearly documenting the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
 - d) make other distribution arrangements with the Copyright Holder.
5. You may charge a reasonable copying fee for any distribution of this

Package. You may charge any fee you choose for support of this Package.
 YOU MAY NOT CHARGE A FEE FOR THIS PACKAGE ITSELF. However, you may
 distribute this Package in aggregate with other (possibly commercial)
 programs as part of a larger (possibly commercial) software distribution
 provided that YOU DO NOT ADVERTISE this package as a product of your
 own.

6. The name of the Copyright Holder may not be used to endorse or
 promote products derived from this software without specific prior
 written permission.

7. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED
 WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF
 MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

/*

* Copyright (c) 1989 - 1994, Julianne Frances Haugh

* Copyright (c) 1996 - 2000, Marek Michałkiewicz

* Copyright (c) 2001 - 2006, Tomasz Kłoczko

* Copyright (c) 2007 - 2011, Nicolas François

* All rights reserved.

*

* Redistribution and use in source and binary forms, with or without

* modification, are permitted provided that the following conditions

* are met:

* 1. Redistributions of source code must retain the above copyright

* notice, this list of conditions and the following disclaimer.

* 2. Redistributions in binary form must reproduce the above copyright

* notice, this list of conditions and the following disclaimer in the

* documentation and/or other materials provided with the distribution.

* 3. The name of the copyright holders or contributors may not be used to

* endorse or promote products derived from this software without

* specific prior written permission.

*

* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

* ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT

* LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A

* PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT

* HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,

* DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY

* THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE

* OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

*/

#include <config.h>

#ident "\$Id\$"

#include <errno.h>

#include <fcntl.h>

#include <getopt.h>

```

#include <pwd.h>
#include <signal.h>
#include <stdio.h>
#include <sys/types.h>
#ifdef WITH_SELINUX
#include <selinux/selinux.h>
#include <selinux/flask.h>
#include <selinux/av_permissions.h>
#include <selinux/context.h>
#endif/* WITH_SELINUX */
#include <time.h>
#include "defines.h"
#include "getdef.h"
#include "nscd.h"
#include "prototypes.h"
#include "pwauth.h"
#include "pwio.h"
#include "shadowio.h"

/*
 * exit status values
 */
/*@-exitarg@*/
#define E_SUCCESS0/* success */
#define E_NOPERM1/* permission denied */
#define E_USAGE2/* invalid combination of options */
#define E_FAILURE3/* unexpected failure, nothing done */
#define E_MISSING4/* unexpected failure, passwd file missing */
#define E_PWDBUSY5/* passwd file busy, try again later */
#define E_BAD_ARG6/* invalid argument to option */
/*
 * Global variables
 */
const char *Prog;/* Program name */

static char *name;/* The name of user whose password is being changed */
static char *myname;/* The current user's name */
static bool amroot;/* The caller's real UID was 0 */

static bool
    aflag = false;/* -a - show status for all users */
    dflag = false;/* -d - delete password */
    eflag = false;/* -e - force password change */
    iflag = false;/* -i - set inactive days */
    kflag = false;/* -k - change only if expired */
    lflag = false;/* -l - lock the user's password */
    nflag = false;/* -n - set minimum days */
    qflag = false;/* -q - quiet mode */
    Sflag = false;/* -S - show password status */
    uflag = false;/* -u - unlock the user's password */
    wflag = false;/* -w - set warning days */
    xflag = false;/* -x - set maximum days */

/*

```

```

* set to 1 if there are any flags which require root privileges,
* and require username to be specified
*/
static bool anyflag = false;

static long age_min = 0; /* Minimum days before change */
static long age_max = 0; /* Maximum days until change */
static long warn = 0; /* Warning days before change */
static long inact = 0; /* Days without change before locked */

#ifndef USE_PAM
static bool do_update_age = false;
#endif /* !USE_PAM */

static bool pw_locked = false;
static bool spw_locked = false;

#ifndef USE_PAM
/*
* Size of the biggest passwd:
* $6$3
* rounds=7
* 9999999999
* $1
* salt16
* $1
* SHA512123
* null
*
* total161
*/
static char crypt_passwd[256];
static bool do_update_pwd = false;
#endif /* !USE_PAM */

/*
* External identifiers
*/

/* local function prototypes */
static /*@noreturn@*/void usage (int);

#ifndef USE_PAM
static bool reuse (const char *, const struct passwd *);
static int new_password (const struct passwd *);

static void check_password (const struct passwd *, const struct spwd *);
#endif /* !USE_PAM */
static /*@observer@*/const char *date_to_str (time_t);
static /*@observer@*/const char *pw_status (const char *);
static void print_status (const struct passwd *);
static /*@noreturn@*/void fail_exit (int);
static /*@noreturn@*/void oom (void);
static char *update_crypt_pw (char *);
static void update_noshadow (void);

```



```

static void update_shadow (void);
#ifdef WITH_SELINUX
static int check_selinux_access (const char *changed_user,
                                uid_t changed_uid,
                                access_vector_t requested_access);
#endif/* WITH_SELINUX */

/*
 * usage - print command usage and exit
 */
static /*@noreturn@*/void usage (int status)
{
FILE *usageout = (E_SUCCESS != status) ? stderr : stdout;
(void) fprintf (usageout,
                _("Usage: %s [options] [LOGIN]\n"
                  "\n"
                  "Options:\n"),
                Prog);
(void) fputs _(" -a, --all          report password status on all accounts\n"), usageout);
(void) fputs _(" -d, --delete       delete the password for the named account\n"), usageout);
(void) fputs _(" -e, --expire       force expire the password for the named account\n"), usageout);
(void) fputs _(" -h, --help         display this help message and exit\n"), usageout);
(void) fputs _(" -k, --keep-tokens   change password only if expired\n"), usageout);
(void) fputs _(" -i, --inactive INACTIVE set password inactive after expiration\n"
                "                  to INACTIVE\n"), usageout);
(void) fputs _(" -l, --lock          lock the password of the named account\n"), usageout);
(void) fputs _(" -n, --mindays MIN_DAYS set minimum number of days before password\n"
                "                  change to MIN_DAYS\n"), usageout);
(void) fputs _(" -q, --quiet        quiet mode\n"), usageout);
(void) fputs _(" -r, --repository REPOSITORY change password in REPOSITORY repository\n"), usageout);
(void) fputs _(" -R, --root CHROOT_DIR directory to chroot into\n"), usageout);
(void) fputs _(" -S, --status        report password status on the named account\n"), usageout);
(void) fputs _(" -u, --unlock        unlock the password of the named account\n"), usageout);
(void) fputs _(" -w, --warndays WARN_DAYS set expiration warning days to WARN_DAYS\n"), usageout);
(void) fputs _(" -x, --maxdays MAX_DAYS set maximum number of days before password\n"
                "                  change to MAX_DAYS\n"), usageout);
(void) fputs ("\n", usageout);
exit (status);
}

#ifdef USE_PAM
static bool reuse (const char *pass, const struct passwd *pw)
{
#ifdef HAVE_LIBCRACK_HIST
const char *reason;

#ifdef HAVE_LIBCRACK_PW
const char *FascistHistoryPw (const char *, const struct passwd *);

reason = FascistHistory (pass, pw);
#else/* !HAVE_LIBCRACK_PW */
const char *FascistHistory (const char *, int);

```

```

reason = FascistHistory (pass, pw->pw_uid);
#endif/* !HAVE_LIBCRACK_PW */
if (NULL != reason) {
(void) printf (_("Bad password: %s. "), reason);
return true;
}
#endif/* HAVE_LIBCRACK_HIST */
return false;
}

/*
 * new_password - validate old password and replace with new (both old and
 * new in global "char crypt_passwd[128]")
 */
static int new_password (const struct passwd *pw)
{
char *clear;/* Pointer to clear text */
char *cipher;/* Pointer to cipher text */
const char *salt;/* Pointer to new salt */
char *cp;/* Pointer to getpass() response */
char orig[200];/* Original password */
char pass[200];/* New password */
int i;/* Counter for retries */
bool warned;
int pass_max_len = -1;
const char *method;

#ifdef HAVE_LIBCRACK_HIST
int HistUpdate (const char *, const char *);
#endif/* HAVE_LIBCRACK_HIST */

/*
 * Authenticate the user. The user will be prompted for their own
 * password.
 */

if (!amroot && ('\0' != crypt_passwd[0])) {
clear = getpass (_("Old password: "));
if (NULL == clear) {
return -1;
}

cipher = pw_encrypt (clear, crypt_passwd);

if (NULL == cipher) {
strzero (clear);
fprintf (stderr,
_(""%s: failed to crypt password with previous salt: %s\n"),
Prog, strerror (errno));
SYSLOG ((LOG_INFO,
"Failed to crypt password with previous salt of user %s",
pw->pw_name));
return -1;
}
}

```

```

if (strcmp (cipher, crypt_passwd) != 0) {
    strzero (clear);
    strzero (cipher);
    SYSLOG ((LOG_WARN, "incorrect password for %s",
        pw->pw_name));
    (void) sleep (1);
    (void) fprintf (stderr,
        _("Incorrect password for %s.\n"),
        pw->pw_name);
    return -1;
}
STRFCPY (orig, clear);
strzero (clear);
strzero (cipher);
} else {
    orig[0] = '\0';
}

/*
 * Get the new password. The user is prompted for the new password
 * and has five tries to get it right. The password will be tested
 * for strength, unless it is the root user. This provides an escape
 * for initial login passwords.
 */
method = getdef_str ("ENCRYPT_METHOD");
if (NULL == method) {
    if (!getdef_bool ("MD5_CRYPT_ENAB")) {
        pass_max_len = getdef_num ("PASS_MAX_LEN", 8);
    }
    } else {
        if ( (strcmp (method, "MD5") == 0)
#ifdef USE_SHA_CRYPT
            || (strcmp (method, "SHA256") == 0)
            || (strcmp (method, "SHA512") == 0)
#endif
        ) {
            pass_max_len = -1;
        } else {
            pass_max_len = getdef_num ("PASS_MAX_LEN", 8);
        }
    }
    if (!qflag) {
        if (pass_max_len == -1) {
            (void) printf (_(
                "Enter the new password (minimum of %d characters)\n"
                "Please use a combination of upper and lower case letters and numbers.\n"),
                getdef_num ("PASS_MIN_LEN", 5));
        } else {
            (void) printf (_(
                "Enter the new password (minimum of %d, maximum of %d characters)\n"
                "Please use a combination of upper and lower case letters and numbers.\n"),
                getdef_num ("PASS_MIN_LEN", 5), pass_max_len);
        }
    }
}

```

```

warned = false;
for (i = getdef_num ("PASS_CHANGE_TRIES", 5); i > 0; i--) {
cp = getpass (_("New password: "));
if (NULL == cp) {
memzero (orig, sizeof orig);
return -1;
}
if (warned && (strcmp (pass, cp) != 0)) {
warned = false;
}
STRFCPY (pass, cp);
strzero (cp);

if (!amroot && (!obscure (orig, pass, pw) || reuse (pass, pw))) {
(void) puts (_("Try again."));
continue;
}

/*
 * If enabled, warn about weak passwords even if you are
 * root (enter this password again to use it anyway).
 * --marekm
 */
if (amroot && !warned && getdef_bool ("PASS_ALWAYS_WARN")
    && (!obscure (orig, pass, pw) || reuse (pass, pw))) {
(void) puts (_("\nWarning: weak password (enter it again to use it anyway)."));
warned = true;
continue;
}
cp = getpass (_("Re-enter new password: "));
if (NULL == cp) {
memzero (orig, sizeof orig);
return -1;
}
if (strcmp (cp, pass) != 0) {
(void) fputs (_("They don't match; try again.\n"), stderr);
} else {
strzero (cp);
break;
}
}
memzero (orig, sizeof orig);

if (i == 0) {
memzero (pass, sizeof pass);
return -1;
}

/*
 * Encrypt the password, then wipe the cleartext password.
 */
salt = crypt_make_salt (NULL, NULL);
cp = pw_encrypt (pass, salt);
memzero (pass, sizeof pass);

```

```

if (NULL == cp) {
    fprintf (stderr,
        _("%s: failed to crypt password with salt '%s': %s\n"),
        Prog, salt, strerror (errno));
    return -1;
}

#ifdef HAVE_LIBCRACK_HIST
HistUpdate (pw->pw_name, crypt_passwd);
#endif /* HAVE_LIBCRACK_HIST */
STRFCPY (crypt_passwd, cp);
return 0;
}

/*
 * check_password - test a password to see if it can be changed
 *
 * check_password() sees if the invoker has permission to change the
 * password for the given user.
 */
static void check_password (const struct passwd *pw, const struct spwd *sp)
{
    time_t now;
    int exp_status;

    exp_status = isexpired (pw, sp);

    /*
     * If not expired and the "change only if expired" option (idea from
     * PAM) was specified, do nothing. --marekm
     */
    if (kflag && (0 == exp_status)) {
        exit (E_SUCCESS);
    }

    /*
     * Root can change any password any time.
     */
    if (amroot) {
        return;
    }

    (void) time (&now);

    /*
     * Expired accounts cannot be changed ever. Passwords which are
     * locked may not be changed. Passwords where min > max may not be
     * changed. Passwords which have been inactive too long cannot be
     * changed.
     */
    if ( (sp->sp_pwdp[0] == '!')
        || (exp_status > 1)
        || ( (sp->sp_max >= 0)

```

```

        && (sp->sp_min > sp->sp_max))) {
(void) fprintf (stderr,
                _("The password for %s cannot be changed.\n"),
                sp->sp_namp);
SYSLOG ((LOG_WARN, "password locked for '%s'", sp->sp_namp));
closelog ();
exit (E_NOPERM);
}

/*
 * Passwords may only be changed after sp_min time is up.
 */
if (sp->sp_lstchg > 0) {
time_t ok;
ok = (time_t) sp->sp_lstchg * SCALE;
if (sp->sp_min > 0) {
ok += (time_t) sp->sp_min * SCALE;
}

if (now < ok) {
(void) fprintf (stderr,
                _("The password for %s cannot be changed yet.\n"),
                pw->pw_name);
SYSLOG ((LOG_WARN, "now < minimum age for '%s'", pw->pw_name));
closelog ();
exit (E_NOPERM);
}
}
}
#endif/* !USE_PAM */

static /*@observer@*/const char *date_to_str (time_t t)
{
static char buf[80];
struct tm *tm;

tm = gmtime (&t);
#ifdef HAVE_STRFTIME
(void) strftime (buf, sizeof buf, "%m/%d/%Y", tm);
#else/* !HAVE_STRFTIME */
(void) snprintf (buf, sizeof buf, "%02d/%02d/%04d",
                tm->tm_mon + 1, tm->tm_mday, tm->tm_year + 1900);
#endif/* !HAVE_STRFTIME */
return buf;
}

static /*@observer@*/const char *pw_status (const char *pass)
{
{
if (*pass == '*' || *pass == '!') {
return "L";
}
if (*pass == '\0') {
return "NP";
}
}
return "P";
}

```

```

}

/*
 * print_status - print current password status
 */
static void print_status (const struct passwd *pw)
{
    struct spwd *sp;

    sp = getspnam (pw->pw_name); /* local, no need for xgetspnam */
    if (NULL != sp) {
        (void) printf ("%s %s %s %lld %lld %lld %lld\n",
            pw->pw_name,
            pw_status (sp->sp_pwdp),
            date_to_str (sp->sp_lstchg * SCALE),
            ((long long)sp->sp_min * SCALE) / DAY,
            ((long long)sp->sp_max * SCALE) / DAY,
            ((long long)sp->sp_warn * SCALE) / DAY,
            ((long long)sp->sp_inact * SCALE) / DAY);
    } else {
        (void) printf ("%s %s\n",
            pw->pw_name, pw_status (pw->pw_passwd));
    }
}

static /*@noreturn@*/void fail_exit (int status)
{
    if (pw_locked) {
        if (pw_unlock () == 0) {
            (void) fprintf (stderr, _("%s: failed to unlock %s\n"), Prog, pw_dbname ());
            SYSLOG ((LOG_ERR, "failed to unlock %s", pw_dbname ());
            /* continue */
        }
    }
}

if (spw_locked) {
    if (spw_unlock () == 0) {
        (void) fprintf (stderr, _("%s: failed to unlock %s\n"), Prog, spw_dbname ());
        SYSLOG ((LOG_ERR, "failed to unlock %s", spw_dbname ());
        /* continue */
    }
}

exit (status);
}

static /*@noreturn@*/void oom (void)
{
    (void) fprintf (stderr, _("%s: out of memory\n"), Prog);
    fail_exit (E_FAILURE);
}

static char *update_crypt_pw (char *cp)

```

```

{
#ifdef USE_PAM
if (do_update_pwd) {
cp = xstrdup (crypt_passwd);
}
#endif/* !USE_PAM */

if (dflg) {
*cp = '\0';
}

if (uflg && *cp == '!') {
if (cp[1] == '\0') {
(void) fprintf (stderr,
_("%s: unlocking the password would result in a passwordless account.\n"
"You should set a password with usermod -p to unlock the password of this account.\n"),
Prog);
fail_exit (E_FAILURE);
} else {
cp++;
}
}

if (lflg && *cp != '!') {
char *newpw = xmalloc (strlen (cp) + 2);

strcpy (newpw, "!");
strcat (newpw, cp);
cp = newpw;
}
return cp;
}

static void update_noshadow (void)
{
const struct passwd *pw;
struct passwd *npw;

if (pw_lock () == 0) {
(void) fprintf (stderr,
_("%s: cannot lock %s; try again later.\n"),
Prog, pw_dbname ());
exit (E_PWDBUSY);
}
pw_locked = true;
if (pw_open (O_RDWR) == 0) {
(void) fprintf (stderr,
_("%s: cannot open %s\n"),
Prog, pw_dbname ());
SYSLOG ((LOG_WARN, "cannot open %s", pw_dbname ()));
fail_exit (E_MISSING);
}
pw = pw_locate (name);
if (NULL == pw) {

```



```

(void) fprintf (stderr,
                _("%s: user '%s' does not exist in %s\n"),
                Prog, name, pw_dbname ());
fail_exit (E_NOPERM);
}
npw = __pw_dup (pw);
if (NULL == npw) {
oom ();
}
npw->pw_passwd = update_crypt_pw (npw->pw_passwd);
if (pw_update (npw) == 0) {
(void) fprintf (stderr,
                _("%s: failed to prepare the new %s entry '%s'\n"),
                Prog, pw_dbname (), npw->pw_name);
fail_exit (E_FAILURE);
}
if (pw_close () == 0) {
(void) fprintf (stderr,
                _("%s: failure while writing changes to %s\n"),
                Prog, pw_dbname ());
SYSLOG ((LOG_ERR, "failure while writing changes to %s", pw_dbname ());
fail_exit (E_FAILURE);
}
if (pw_unlock () == 0) {
(void) fprintf (stderr,
                _("%s: failed to unlock %s\n"),
                Prog, pw_dbname ());
SYSLOG ((LOG_ERR, "failed to unlock %s", pw_dbname ());
/* continue */
}
pw_locked = false;
}

static void update_shadow (void)
{
const struct spwd *sp;
struct spwd *nsp;

if (spw_lock () == 0) {
(void) fprintf (stderr,
                _("%s: cannot lock %s; try again later.\n"),
                Prog, spw_dbname ());
exit (E_PWDBUSY);
}
spw_locked = true;
if (spw_open (O_RDWR) == 0) {
(void) fprintf (stderr,
                _("%s: cannot open %s\n"),
                Prog, spw_dbname ());
SYSLOG ((LOG_WARN, "cannot open %s", spw_dbname ());
fail_exit (E_FAILURE);
}
sp = spw_locate (name);
if (NULL == sp) {

```

```

/* Try to update the password in /etc/passwd instead. */
(void) spw_close ();
update_noshadow ();
if (spw_unlock () == 0) {
(void) fprintf (stderr,
    _("%s: failed to unlock %s\n"),
    Prog, spw_dbname ());
SYSLOG ((LOG_ERR, "failed to unlock %s", spw_dbname ());
/* continue */
}
spw_locked = false;
return;
}
nsp = __spw_dup (sp);
if (NULL == nsp) {
oom ();
}
nsp->sp_pwdp = update_crypt_pw (nsp->sp_pwdp);
if (xflg) {
nsp->sp_max = (age_max * DAY) / SCALE;
}
if (nflg) {
nsp->sp_min = (age_min * DAY) / SCALE;
}
if (wflg) {
nsp->sp_warn = (warn * DAY) / SCALE;
}
if (iflg) {
nsp->sp_inact = (inact * DAY) / SCALE;
}
#ifdef USE_PAM
if (do_update_age) {
nsp->sp_lstchg = (long) time ((time_t *) 0) / SCALE;
if (0 == nsp->sp_lstchg) {
/* Better disable aging than requiring a password
 * change */
nsp->sp_lstchg = -1;
}
}
#endif /* !USE_PAM */

/*
 * Force change on next login, like SunOS 4.x passwd -e or Solaris
 * 2.x passwd -f. Solaris 2.x seems to do the same thing (set
 * sp_lstchg to 0).
 */
if (eflg) {
nsp->sp_lstchg = 0;
}

if (spw_update (nsp) == 0) {
(void) fprintf (stderr,
    _("%s: failed to prepare the new %s entry %s\n"),
    Prog, spw_dbname (), nsp->sp_namp);
fail_exit (E_FAILURE);

```

```

}
if (spw_close () == 0) {
(void) fprintf (stderr,
_("%s: failure while writing changes to %s\n"),
Prog, spw_dbname ());
SYSLOG ((LOG_ERR, "failure while writing changes to %s", spw_dbname ());
fail_exit (E_FAILURE);
}
if (spw_unlock () == 0) {
(void) fprintf (stderr,
_("%s: failed to unlock %s\n"),
Prog, spw_dbname ());
SYSLOG ((LOG_ERR, "failed to unlock %s", spw_dbname ());
/* continue */
}
spw_locked = false;
}

#ifdef WITH_SELINUX
static int check_selinux_access (const char *changed_user,
uid_t changed_uid,
access_vector_t requested_access)
{
int status = -1;
security_context_t user_context;
context_t c;
const char *user;

/* if in permissive mode then allow the operation */
if (security_getenforce() == 0) {
return 0;
}

/* get the context of the process which executed passwd */
if (getprevcon(&user_context) != 0) {
return -1;
}

/* get the "user" portion of the context (the part before the first
colon) */
c = context_new(user_context);
user = context_user_get(c);

/* if changing a password for an account with UID==0 or for an account
where the identity matches then return success */
if (changed_uid != 0 && strcmp(changed_user, user) == 0) {
status = 0;
} else {
struct av_decision avd;
int retval;
retval = security_compute_av(user_context,
user_context,
SECCLASS_PASSWD,
requested_access,

```

```

        &avd);
if ((retval == 0) &&
    ((requested_access & avd.allowed) == requested_access)) {
    status = 0;
}
}
context_free(c);
freecon(user_context);
return status;
}

#endif/* WITH_SELINUX */

/*
 * passwd - change a user's password file information
 *
 * This command controls the password file and commands which are used
 * to modify it.
 *
 * The valid options are
 *
 * -ddelete the password for the named account (*)
 * -eexpire the password for the named account (*)
 * -fexecute chfn command to interpret flags
 * -gexecute gpasswd command to interpret flags
 * -i #set sp_inact to # days (*)
 * -kchange password only if expired
 * -llock the password of the named account (*)
 * -n #set sp_min to # days (*)
 * -r #change password in # repository
 * -sexecute chsh command to interpret flags
 * -Sshow password status of named account
 * -uunlock the password of the named account (*)
 * -w #set sp_warn to # days (*)
 * -x #set sp_max to # days (*)
 *
 * (*) requires root permission to execute.
 *
 * All of the time fields are entered in days and converted to the
 * appropriate internal format. For finer resolute the chage
 * command must be used.
 */
int main (int argc, char **argv)
{
    const struct passwd *pw;/* Password file entry for user */

#ifdef USE_PAM
    char *cp;/* Miscellaneous character pointing */

    const struct spwd *sp;/* Shadow file entry for user */
#endif/* !USE_PAM */

    sanitize_env ();

    /*

```

```

* Get the program name. The program name is used as a prefix to
* most error messages.
*/
Prog = Basename (argv[0]);

(void) setlocale (LC_ALL, "");
(void) bindtextdomain (PACKAGE, LOCALEDIR);
(void) textdomain (PACKAGE);

process_root_flag ("-R", argc, argv);

/*
* The program behaves differently when executed by root than when
* executed by a normal user.
*/
amroot = (getuid () == 0);

OPENLOG ("passwd");

{
/*
* Parse the command line options.
*/
int c;
static struct option long_options[] = {
{"all",      no_argument,  NULL, 'a'},
{"delete",   no_argument,  NULL, 'd'},
{"expire",   no_argument,  NULL, 'e'},
{"help",     no_argument,  NULL, 'h'},
{"inactive", required_argument, NULL, 'i'},
{"keep-tokens", no_argument,  NULL, 'k'},
{"lock",     no_argument,  NULL, 'l'},
{"mindays",  required_argument, NULL, 'n'},
{"quiet",    no_argument,  NULL, 'q'},
{"repository", required_argument, NULL, 'r'},
{"root",     required_argument, NULL, 'R'},
{"status",   no_argument,  NULL, 'S'},
{"unlock",   no_argument,  NULL, 'u'},
{"warndays", required_argument, NULL, 'w'},
{"maxdays", required_argument, NULL, 'x'},
{NULL, 0, NULL, '\0'}
};

while ((c = getopt_long (argc, argv, "adehi:kl:n:qr:R:Suw:x:",
                        long_options, NULL)) != -1) {
switch (c) {
case 'a':
aflag = true;
break;
case 'd':
dflag = true;
anyflag = true;
break;
case 'e':

```

```

eflg = true;
anyflag = true;
break;
case 'h':
usage (E_SUCCESS);
/*@notreached@*/break;
case 'i':
if ( (getlong (optarg, &inact) == 0)
    || (inact < -1)) {
fprintf (stderr,
        _("%s: invalid numeric argument '%s'\n"),
        Prog, optarg);
usage (E_BAD_ARG);
}
iflg = true;
anyflag = true;
break;
case 'k':
/* change only if expired, like Linux-PAM passwd -k. */
kflg = true; /* ok for users */
break;
case 'l':
lflg = true;
anyflag = true;
break;
case 'n':
if ( (getlong (optarg, &age_min) == 0)
    || (age_min < -1)) {
fprintf (stderr,
        _("%s: invalid numeric argument '%s'\n"),
        Prog, optarg);
usage (E_BAD_ARG);
}
nflg = true;
anyflag = true;
break;
case 'q':
qflg = true; /* ok for users */
break;
case 'r':
/* -r repository (files|nis|nisplus) */
/* only "files" supported for now */
if (strcmp (optarg, "files") != 0) {
fprintf (stderr,
        _("%s: repository %s not supported\n"),
        Prog, optarg);
exit (E_BAD_ARG);
}
break;
case 'R': /* no-op, handled in process_root_flag () */
break;
case 'S':
Sflg = true; /* ok for users */
break;
case 'u':

```

```

    uflag = true;
    anyflag = true;
    break;
case 'w':
    if ( (getlong (optarg, &warn) == 0)
        || (warn < -1)) {
        (void) fprintf (stderr,
            _("%s: invalid numeric argument '%s'\n"),
                Prog, optarg);
        usage (E_BAD_ARG);
    }
    wflag = true;
    anyflag = true;
    break;
case 'x':
    if ( (getlong (optarg, &age_max) == 0)
        || (age_max < -1)) {
        (void) fprintf (stderr,
            _("%s: invalid numeric argument '%s'\n"),
                Prog, optarg);
        usage (E_BAD_ARG);
    }
    xflag = true;
    anyflag = true;
    break;
default:
    usage (E_BAD_ARG);
}
}
}

/*
 * Now I have to get the user name. The name will be gotten from the
 * command line if possible. Otherwise it is figured out from the
 * environment.
 */
pw = get_my_pwent ();
if (NULL == pw) {
    (void) fprintf (stderr,
        _("%s: Cannot determine your user name.\n"),
            Prog);
    SYSLOG ((LOG_WARN, "Cannot determine the user name of the caller (UID %lu)",
        (unsigned long) getuid ());
    exit (E_NOPERM);
}
myname = xstrdup (pw->pw_name);
if (optind < argc) {
    name = argv[optind];
} else {
    name = myname;
}

/*
 * Make sure that at most one username was specified.

```

```

    */
    if (argc > (optind+1)) {
        usage (E_USAGE);
    }

    /*
     * The -a flag requires -S, no other flags, no username, and
     * you must be root. --marekm
     */
    if (aflag) {
        if (anyflag || !Sflag || (optind < argc)) {
            usage (E_USAGE);
        }
        if (!amroot) {
            (void) fprintf (stderr,
                _("%s: Permission denied.\n"),
                Prog);
            exit (E_NOPERM);
        }
        setpwent ();
        while ( (pw = getpwent ()) != NULL ) {
            print_status (pw);
        }
        endpwent ();
        exit (E_SUCCESS);
    }
    #if 0
    /*
     * Allow certain users (administrators) to change passwords of
     * certain users. Not implemented yet. --marekm
     */
    if (may_change_passwd (myname, name))
        amroot = 1;
    #endif

    /*
     * If any of the flags were given, a user name must be supplied on
     * the command line. Only an unadorned command line doesn't require
     * the user's name be given. Also, -x, -n, -w, -i, -e, -d,
     * -l, -u may appear with each other. -S, -k must appear alone.
     */

    /*
     * -S now ok for normal users (check status of my own account), and
     * doesn't require username. --marekm
     */
    if (anyflag && optind >= argc) {
        usage (E_USAGE);
    }

    if ( (Sflag && kflag)
        || (anyflag && (Sflag || kflag))) {
        usage (E_USAGE);
    }

```



```

if (anyflag && !amroot) {
(void) fprintf(stderr, _("%s: Permission denied.\n"), Prog);
exit (E_NOPERM);
}

pw = xgetpwnam (name);
if (NULL == pw) {
(void) fprintf (stderr,
_("%s: user '%s' does not exist\n"),
Prog, name);
exit (E_NOPERM);
}
#ifdef WITH_SELINUX
/* only do this check when getuid()==0 because it's a pre-condition for
changing a password without entering the old one */
if ((is_selinux_enabled() > 0) && (getuid() == 0) &&
(check_selinux_access (name, pw->pw_uid, PASSWD__PASSWD) != 0)) {
security_context_t user_context = NULL;
const char *user = "Unknown user context";
if (getprevcon (&user_context) == 0) {
user = user_context; /* FIXME: use context_user_get? */
}
SYSLOG ((LOG_ALERT,
"%s is not authorized to change the password of %s",
user, name));
(void) fprintf(stderr,
_("%s: %s is not authorized to change the password of %s\n"),
Prog, user, name);
if (NULL != user_context) {
freecon (user_context);
}
exit (E_NOPERM);
}
#endif /* WITH_SELINUX */

/*
* If the UID of the user does not match the current real UID,
* check if I'm root.
*/
if (!amroot && (pw->pw_uid != getuid ())) {
(void) fprintf (stderr,
_("%s: You may not view or modify password information for %s.\n"),
Prog, name);
SYSLOG ((LOG_WARN,
"%s: can't view or modify password information for %s",
Prog, name));
closelog ();
exit (E_NOPERM);
}

if (Sflag) {
print_status (pw);
exit (E_SUCCESS);
}

```

```

#ifndef USE_PAM
/*
 * The user name is valid, so let's get the shadow file entry.
 */
sp = getsppnam (name); /* !USE_PAM, no need for xgetsppnam */
if (NULL == sp) {
if (errno == EACCES) {
(void) fprintf (stderr,
                _("'%s': Permission denied.\n"),
                Prog);
exit (E_NOOPERM);
}
sp = passwd_to_spwd (pw);
}

cp = sp->sp_passwd;

/*
 * If there are no other flags, just change the password.
 */
if (!anyflag) {
STRFCPY (crypt_passwd, cp);

/*
 * See if the user is permitted to change the password.
 * Otherwise, go ahead and set a new password.
 */
check_password (pw, sp);

/*
 * Let the user know whose password is being changed.
 */
if (!qflag) {
(void) printf (_("Changing password for %s\n"), name);
}

if (new_password (pw) != 0) {
(void) fprintf (stderr,
                _("The password for %s is unchanged.\n"),
                name);
closelog ();
exit (E_NOOPERM);
}
do_update_passwd = true;
do_update_age = true;
}
#endif /* !USE_PAM */
/*
 * Before going any further, raise the ulimit to prevent colliding
 * into a lowered ulimit, and set the real UID to root to protect
 * against unexpected signals. Any keyboard signals are set to be
 * ignored.
 */
passwd_init ();

```

```
#ifdef USE_PAM
/*
 * Don't set the real UID for PAM...
 */
if (!anyflag) {
do_pam_passwd (name, qflag, kflag);
exit (E_SUCCESS);
}
#endif/* USE_PAM */
if (setuid (0) != 0) {
(void) fputs (_("Cannot change ID to root.\n"), stderr);
SYSLOG ((LOG_ERR, "can't setuid(0)"));
closelog ();
exit (E_NOPERM);
}
if (spw_file_present ()) {
update_shadow ();
} else {
update_noshadow ();
}

nscd_flush_cache ("passwd");
nscd_flush_cache ("group");

SYSLOG ((LOG_INFO, "password for '%s' changed by '%s'", name, myname));
closelog ();
if (!qflag) {
if (!anyflag) {
#ifdef USE_PAM
(void) printf (_("%s: password changed.\n"), Prog);
#endif/* USE_PAM */
} else {
(void) printf (_("%s: password expiry information changed.\n"), Prog);
}
}

return E_SUCCESS;
}
```

1.126 shellinabox

1.126.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder say-

ing it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geograph-

ical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w`. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c`
for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.127 slang

1.127.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.128 strace

1.128.1 Available under license

Copyright (c) 1991, 1992 Paul Kranenburg <pk@cs.few.eur.nl>
Copyright (c) 1993 Branko Lankester <branko@hacktic.nl>
Copyright (c) 1993 Ulrich Pegelow <pegelow@moorea.uni-muenster.de>
Copyright (c) 1995, 1996 Michael Elizabeth Chastain <mec@duracef.shout.net>
Copyright (c) 1993, 1994, 1995, 1996 Rick Sladkey <jrs@world.std.com>
Copyright (C) 1998-2001 Wichert Akkerman <wakkerma@deephackmode.org>
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.129 sudo

1.129.1 Available under license

```
/*$OpenBSD: fnmatch.c,v 1.15 2011/02/10 21:31:59 stsp Exp $*/
```

```
/* Copyright (c) 2011, VMware, Inc.
```

```
* All rights reserved.
```

```
*
```

```
* Redistribution and use in source and binary forms, with or without
```

```
* modification, are permitted provided that the following conditions are met:
```

```
* * Redistributions of source code must retain the above copyright
```

```
* notice, this list of conditions and the following disclaimer.
```

```
* * Redistributions in binary form must reproduce the above copyright
```

```
* notice, this list of conditions and the following disclaimer in the
```

```
* documentation and/or other materials provided with the distribution.
```

```
* * Neither the name of the VMware, Inc. nor the names of its contributors
```

```
* may be used to endorse or promote products derived from this software
```

```
* without specific prior written permission.
```

```
*
```

```
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
```

```
* AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
```

```
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
```

```
* ARE DISCLAIMED. IN NO EVENT SHALL VMWARE, INC. OR CONTRIBUTORS BE LIABLE FOR
```

```
* ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
```

```
* (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
```

```
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
```

```
* ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
```

```
* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
```

```
* THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

```
*/
```

```
/*
```

```
* Copyright (c) 2008, 2016 Todd C. Miller <millert@openbsd.org>
```

```
*
```

```
* Permission to use, copy, modify, and distribute this software for any
```

```
* purpose with or without fee is hereby granted, provided that the above
```

```
* copyright notice and this permission notice appear in all copies.
```

```
*
```

```
* THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
```

```
* WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
```

```
* MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
```

```
* ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
```

```
* WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
```

```
* ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
```

```
* OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
```

```
*/
```

```
/* Authored by William A. Rowe Jr. <wrowe; apache.org, vmware.com>, April 2011
```

```
*
```

```
* Derived from The Open Group Base Specifications Issue 7, IEEE Std 1003.1-2008
```

```

* as described in;
* http://pubs.opengroup.org/onlinepubs/9699919799/functions/fnmatch.html
*
* Filename pattern matches defined in section 2.13, "Pattern Matching Notation"
* from chapter 2, "Shell Command Language"
* http://pubs.opengroup.org/onlinepubs/9699919799/utilities/V3\_chap02.html#tag\_18\_13
* where; 1. A bracket expression starting with an unquoted <circumflex> '^'
* character CONTINUES to specify a non-matching list; 2. an explicit <period> '.'
* in a bracket expression matching list, e.g. "[.abc]" does NOT match a leading
* <period> in a filename; 3. a <left-square-bracket> '[' which does not introduce
* a valid bracket expression is treated as an ordinary character; 4. a differing
* number of consecutive slashes within pattern and string will NOT match;
* 5. a trailing '\' in FNM_ESCAPE mode is treated as an ordinary '\' character.
*
* Bracket expansion defined in section 9.3.5, "RE Bracket Expression",
* from chapter 9, "Regular Expressions"
* http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1\_chap09.html#tag\_09\_03\_05
* with no support for collating symbols, equivalence class expressions or
* character class expressions. A partial range expression with a leading
* hyphen following a valid range expression will match only the ordinary
* <hyphen> and the ending character (e.g. "[a-m-z]" will match characters
* 'a' through 'm', a <hyphen> '-', or a 'z').
*
* Supports BSD extensions FNM_LEADING_DIR to match pattern to the end of one
* path segment of string, and FNM_CASEFOLD to ignore alpha case.
*
* NOTE: Only POSIX/C single byte locales are correctly supported at this time.
* Notably, non-POSIX locales with FNM_CASEFOLD produce undefined results,
* particularly in ranges of mixed case (e.g. "[A-z]") or spanning alpha and
* nonalpha characters within a range.
*
* XXX comments below indicate porting required for multi-byte character sets
* and non-POSIX locale collation orders; requires mbr* APIs to track shift
* state of pattern and string (rewinding pattern and string repeatedly).
*
* Certain parts of the code assume 0x00-0x3F are unique with any MBCS (e.g.
* UTF-8, SHIFT-JIS, etc). Any implementation allowing '\' as an alternate
* path delimiter must be aware that 0x5C is NOT unique within SHIFT-JIS.
*/

```

```
#include <config.h>
```

```
#ifndef HAVE_FNMATCH
```

```
#include <sys/types.h>
```

```
#include <stdio.h>
```

```
#include <ctype.h>
```

```
#ifdef HAVE_STRING_H
```

```
# include <string.h>
```

```
#endif /* HAVE_STRING_H */
```

```
#ifdef HAVE_STRINGS_H
```

```
# include <strings.h>
```

```
#endif /* HAVE_STRINGS_H */
```

```

#include "sudo_compat.h"
#include "compat/charclass.h"
#include "compat/fnmatch.h"

#define RANGE_MATCH1
#define RANGE_NOMATCH0
#define RANGE_ERROR(-1)

static int
classmatch(const char *pattern, char test, int foldcase, const char **ep)
{
    const char * const mismatch = pattern;
    const char *colon;
    struct cclass *cc;
    int rval = RANGE_NOMATCH;
    size_t len;

    if (pattern[0] != '[' || pattern[1] != ':') {
        *ep = mismatch;
        return RANGE_ERROR;
    }
    pattern += 2;

    if ((colon = strchr(pattern, ':')) == NULL || colon[1] != ']') {
        *ep = mismatch;
        return RANGE_ERROR;
    }
    *ep = colon + 2;
    len = (size_t)(colon - pattern);

    if (foldcase && strncmp(pattern, "upper:]", 7) == 0)
        pattern = "lower:];";
    for (cc = cclasses; cc->name != NULL; cc++) {
        if (!strncmp(pattern, cc->name, len) && cc->name[len] == '\0') {
            if (cc->isctype((unsigned char)test))
                rval = RANGE_MATCH;
            break;
        }
    }
    if (cc->name == NULL) {
        /* invalid character class, treat as normal text */
        *ep = mismatch;
        rval = RANGE_ERROR;
    }
    return rval;
}

/* Most MBCS/collation/case issues handled here. Wildcard '*' is not handled.
 * EOS '\0' and the FNM_PATHNAME '/' delimiters are not advanced over,
 * however the "\\" sequence is advanced to '/'.
 *
 * Both pattern and string are **char to support pointer increment of arbitrary
 * multibyte characters for the given locale, in a later iteration of this code
 */

```

```

static int fnmatch_ch(const char **pattern, const char **string, int flags)
{
    const char * const mismatch = *pattern;
    const int nocase = !(flags & FNM_CASEFOLD);
    const int escape = !(flags & FNM_NOESCAPE);
    const int slash = !(flags & FNM_PATHNAME);
    int result = FNM_NOMATCH;
    const char *startch;
    int negate;

    if (**pattern == '[')
    {
        ++*pattern;

        /* Handle negation, either leading ! or ^ operators (never both) */
        negate = ((**pattern == '!') || (**pattern == '^'));
        if (negate)
            ++*pattern;

        /* ']' is an ordinary character at the start of the range pattern */
        if (**pattern == ']')
            goto leadingclosebrace;

        while (**pattern)
        {
            if (**pattern == ']') {
                ++*pattern;
                /* XXX: Fix for MBCS character width */
                ++*string;
                return (result ^ negate);
            }

            if (escape && (**pattern == '\\')) {
                ++*pattern;

                /* Patterns must be terminated with ']', not EOS */
                if (!**pattern)
                    break;
            }

            /* Patterns must be terminated with ']' not '/' */
            if (slash && (**pattern == '/'))
                break;

            /* Match character classes. */
            switch (classmatch(*pattern, **string, nocase, pattern)) {
            case RANGE_MATCH:
                result = 0;
                continue;
            case RANGE_NOMATCH:
                /* Valid character class but no match. */
                continue;
            default:
                /* Not a valid character class. */
                break;
            }
        }
    }
}

```

```

    }
    if (**pattern)
        break;

leadingclosebrace:
    /* Look at only well-formed range patterns;
     * "x-]" is not allowed unless escaped ("x-\]")
     * XXX: Fix for locale/MBCS character width
     */
    if (((*pattern)[1] == '-') && ((*pattern)[2] != ']'))
    {
        startch = *pattern;
        *pattern += (escape && ((*pattern)[2] == '\\')) ? 3 : 2;

        /* NOT a properly balanced [expr] pattern, EOS terminated
         * or ranges containing a slash in FNM_PATHNAME mode pattern
         * fall out to the rewind and test '[' literal code path
         */
        if (**pattern || (slash && (**pattern == '/')))
            break;

        /* XXX: handle locale/MBCS comparison, advance by MBCS char width */
        if ((*string >= *startch) && (*string <= **pattern))
            result = 0;
        else if (nocase && (isupper((unsigned char)**string) ||
            isupper((unsigned char)*startch) ||
            isupper((unsigned char)**pattern))
            && (tolower((unsigned char)**string) >= tolower((unsigned char)*startch))
            && (tolower((unsigned char)**string) <= tolower((unsigned char)**pattern)))
            result = 0;

        ++*pattern;
        continue;
    }

    /* XXX: handle locale/MBCS comparison, advance by MBCS char width */
    if ((*string == **pattern))
        result = 0;
    else if (nocase && (isupper((unsigned char)**string) ||
        isupper((unsigned char)**pattern))
        && (tolower((unsigned char)**string) == tolower((unsigned char)**pattern)))
        result = 0;

    ++*pattern;
}

/* NOT a properly balanced [expr] pattern; Rewind
 * and reset result to test '[' literal
 */
*pattern = mismatch;
result = FNM_NOMATCH;
}
else if (**pattern == '?') {
    /* Optimize '?' match before unescaping **pattern */

```

```

        if (!**string || (slash && (**string == '/')))
            return FNM_NOMATCH;
        result = 0;
        goto fnmatch_ch_success;
    }
    else if (escape && (**pattern == '\\') && (*pattern)[1]) {
        ++*pattern;
    }

    /* XXX: handle locale/MBCS comparison, advance by the MBCS char width */
    if (**string == **pattern)
        result = 0;
    else if (nocase && (isupper((unsigned char)**string) || isupper((unsigned char)**pattern))
        && (tolower((unsigned char)**string) == tolower((unsigned char)**pattern)))
        result = 0;

    /* Refuse to advance over trailing slash or nulls
    */
    if (!**string || !**pattern || (slash && (**string == '/') || (**pattern == '/')))
        return result;

fnmatch_ch_success:
    ++*pattern;
    ++*string;
    return result;
}

int sudo_fnmatch(const char *pattern, const char *string, int flags)
{
    static const char dummystring[2] = {' ', 0};
    const int escape = !(flags & FNM_NOESCAPE);
    const int slash = !(flags & FNM_PATHNAME);
    const int leading_dir = !(flags & FNM_LEADING_DIR);
    const char *strendseg;
    const char *dummyptr;
    const char *matchptr;
    int wild;
    /* For '*' wild processing only; suppress 'used before initialization'
    * warnings with dummy initialization values;
    */
    const char *strstartseg = NULL;
    const char *mismatch = NULL;
    int matchlen = 0;

    if (*pattern == '*')
        goto firstsegment;

    while (*pattern && *string)
    {
        /* Pre-decode "\\" which has no special significance, and
        * match balanced slashes, starting a new segment pattern
        */
        if (slash && escape && (*pattern == '\\') && (pattern[1] == '/'))
            ++pattern;
        if (slash && (*pattern == '/') && (*string == '/')) {

```

```

    ++pattern;
    ++string;
}

```

firstsegment:

```

/* At the beginning of each segment, validate leading period behavior.
*/
if ((flags & FNM_PERIOD) && (*string == '.'))
{
    if (*pattern == '.')
        ++pattern;
    else if (escape && (*pattern == '\\') && (pattern[1] == '.'))
        pattern += 2;
    else
        return FNM_NOMATCH;
    ++string;
}

/* Determine the end of string segment
*
* Presumes '/' character is unique, not composite in any MBCS encoding
*/
if (slash) {
    strendseg = strchr(string, '/');
    if (!strendseg)
        strendseg = strchr(string, '\0');
}
else {
    strendseg = strchr(string, '\0');
}

/* Allow pattern '*' to be consumed even with no remaining string to match
*/
while (*pattern)
{
    if ((string > strendseg)
        || ((string == strendseg) && (*pattern != '*')))
        break;

    if (slash && ((*pattern == '/')
        || (escape && (*pattern == '\\')
        && (pattern[1] == '/))))
        break;

    /* Reduce groups of '*' and '?' to n '?' matches
    * followed by one '*' test for simplicity
    */
    for (wild = 0; ((*pattern == '*') || (*pattern == '?')); ++pattern)
    {
        if (*pattern == '*') {
            wild = 1;
        }
        else if (string < strendseg) { /* && (*pattern == '?') */
            /* XXX: Advance 1 char for MBCS locale */

```

```

        ++string;
    }
    else { /* (string >= strendseg) && (*pattern == '?') */
        return FNM_NOMATCH;
    }
}

if (wild)
{
    strstartseg = string;
    mismatch = pattern;

    /* Count fixed (non '*') char matches remaining in pattern
     * excluding '/' (or "\") and '*'
     */
    for (matchptr = pattern, matchlen = 0; 1; ++matchlen)
    {
        if ((*matchptr == '\0')
            || (slash && (*matchptr == '/')
                || (escape && (*matchptr == '\\')
                    && (matchptr[1] == '/'))))
        {
            /* Compare precisely this many trailing string chars,
             * the resulting match needs no wildcard loop
             */
            /* XXX: Adjust for MBCS */
            if (string + matchlen > strendseg)
                return FNM_NOMATCH;

            string = strendseg - matchlen;
            wild = 0;
            break;
        }

        if (*matchptr == '*')
        {
            /* Ensure at least this many trailing string chars remain
             * for the first comparison
             */
            /* XXX: Adjust for MBCS */
            if (string + matchlen > strendseg)
                return FNM_NOMATCH;

            /* Begin first wild comparison at the current position */
            break;
        }

        /* Skip forward in pattern by a single character match
         * Use a dummy fnmatch_ch() test to count one "[range]" escape
         */
        /* XXX: Adjust for MBCS */
        if (escape && (*matchptr == '\\') && matchptr[1]) {
            matchptr += 2;
        }
        else if (*matchptr == '[') {

```



```

        dummyptr = dummystring;
        fnmatch_ch(&matchptr, &dummyptr, flags);
    }
    else {
        ++matchptr;
    }
}

/* Incrementally match string against the pattern
*/
while (*pattern && (string < strendseg))
{
    /* Success; begin a new wild pattern search
    */
    if (*pattern == '*')
        break;

    if (slash && ((*string == '/')
        || (*pattern == '/')
        || (escape && (*pattern == "\\")
            && (pattern[1] == '/'))))
        break;

    /* Compare ch's (the pattern is advanced over "\v" to the '/',
    * but slashes will mismatch, and are not consumed)
    */
    if (!fnmatch_ch(&pattern, &string, flags))
        continue;

    /* Failed to match, loop against next char offset of string segment
    * until not enough string chars remain to match the fixed pattern
    */
    if (wild) {
        /* XXX: Advance 1 char for MBCS locale */
        string = ++strstartseg;
        if (string + matchlen > strendseg)
            return FNM_NOMATCH;

        pattern = mismatch;
        continue;
    }
    else
        return FNM_NOMATCH;
}

if (*string && !(slash || leading_dir) && (*string == '/'))
    return FNM_NOMATCH;

if (*pattern && !(slash && ((*pattern == '/')
    || (escape && (*pattern == "\\")
        && (pattern[1] == '/')))))
    return FNM_NOMATCH;

```

```

        if (leading_dir && !*pattern && *string == '/')
            return 0;
    }

    /* Where both pattern and string are at EOS, declare success
    */
    if (!*string && !*pattern)
        return 0;

    /* pattern didn't match to the end of string */
    return FNM_NOMATCH;
}
#endif /* HAVE_FNMATCH */
/*
 * Copyright (c) 1989, 1991, 1993
 * The Regents of the University of California. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include <config.h>

#ifndef HAVE_GETCWD

#include <sys/types.h>
#include <sys/stat.h>

#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#ifdef HAVE_STRING_H
#include <string.h>

```

```

#endif /* HAVE_STRING_H */
#ifdef HAVE_STRINGS_H
# include <strings.h>
#endif /* HAVE_STRINGS_H */
#include <unistd.h>
#include <dirent.h>

#include "sudo_compat.h"

#define ISDOT(dp) \
(dp->d_name[0] == '.' && (dp->d_name[1] == '\0' || \
(dp->d_name[1] == '.' && dp->d_name[2] == '\0')))

#if defined(HAVE_STRUCT_DIRENT_D_NAMLEN) && HAVE_STRUCT_DIRENT_D_NAMLEN
# define NAMLEN(dirent)(dirent->d_namlen)
#else
# define NAMLEN(dirent)strlen((dirent->d_name)
#endif

char *
sudo_getcwd(char *pt, size_t size)
{
    struct dirent *dp;
    DIR *dir = NULL;
    dev_t dev;
    ino_t ino;
    int first;
    char *bpt, *bup;
    struct stat s;
    dev_t root_dev;
    ino_t root_ino;
    size_t psize, upsize;
    int save_errno;
    char *ept, *eup, *up;

    /*
     * If no buffer specified by the user, allocate one as necessary.
     * If a buffer is specified, the size has to be non-zero. The path
     * is built from the end of the buffer backwards.
     */
    if (pt) {
        psize = 0;
        if (!size) {
            errno = EINVAL;
            return NULL;
        }
        ept = pt + size;
    } else {
        if ((pt = malloc(psize = 1024 - 4)) == NULL)
            return NULL;
        ept = pt + psize;
    }
    bpt = ept - 1;
    *bpt = '\0';

```

```

/*
 * Allocate bytes (1024 - malloc space) for the string of "../"s.
 * Should always be enough (it's 340 levels). If it's not, allocate
 * as necessary. Special * case the first stat, it's ".", not "../".
 */
if ((up = malloc(upsized = 1024 - 4)) == NULL)
goto err;
eup = up + PATH_MAX;
bup = up;
up[0] = '.';
up[1] = '\0';

/* Save root values, so know when to stop. */
if (stat("/", &s))
goto err;
root_dev = s.st_dev;
root_ino = s.st_ino;

errno = 0; /* XXX readdir has no error return. */

for (first = 1;; first = 0) {
/* Stat the current level. */
if (lstat(up, &s))
goto err;

/* Save current node values. */
ino = s.st_ino;
dev = s.st_dev;

/* Check for reaching root. */
if (root_dev == dev && root_ino == ino) {
*--bpt = '/';
/*
 * It's unclear that it's a requirement to copy the
 * path to the beginning of the buffer, but it's always
 * been that way and stuff would probably break.
 */
bcopy(bpt, pt, ept - bpt);
free(up);
return pt;
}

/*
 * Build pointer to the parent directory, allocating memory
 * as necessary. Max length is 3 for "../", the largest
 * possible component name, plus a trailing NULL.
 */
if (bup + 3 + MAXNAMLEN + 1 >= eup) {
char *nup;

if ((nup = reallocarray(up, upsized, 2)) == NULL)
goto err;
upsized *= 2;
up = nup;

```

```
bup = up;
eup = up + upsize;
}
*bup++ = '.';
*bup++ = '.';
*bup = '\0';

/* Open and stat parent directory. */
if (!(dir = opendir(up)) || fstat(dirdir(dir), &s))
goto err;

/* Add trailing slash for next directory. */
*bup++ = '/';

/*
 * If it's a mount point, have to stat each element because
 * the inode number in the directory is for the entry in the
 * parent directory, not the inode number of the mounted file.
 */
save_errno = 0;
if (s.st_dev == dev) {
for (;;) {
if (!(dp = readdir(dir)))
goto notfound;
if (dp->d_ino == ino)
break;
}
} else
for (;;) {
if (!(dp = readdir(dir)))
goto notfound;
if (ISDOT(dp))
continue;
bcopy(dp->d_name, bup, NAMLEN(dp) + 1);

/* Save the first error for later. */
if (lstat(up, &s)) {
if (!save_errno)
save_errno = errno;
errno = 0;
continue;
}
if (s.st_dev == dev && s.st_ino == ino)
break;
}

/*
 * Check for length of the current name, preceding slash,
 * leading slash.
 */
if (bpt - pt <= NAMLEN(dp) + (first ? 1 : 2)) {
size_t len, off;
char *npt;
```

```

    if (!ptsize) {
        errno = ERANGE;
        goto err;
    }
    off = bpt - pt;
    len = ept - bpt;
    if ((npt = reallocarray(pt, psize, 2)) == NULL)
        goto err;
    psize *= 2;
    pt = npt;
    bpt = pt + off;
    ept = pt + psize;
    bcopy(bpt, ept - len, len);
    bpt = ept - len;
}
if (!first)
    *--bpt = '/';
bpt -= NAMLEN(dp);
bcopy(dp->d_name, bpt, NAMLEN(dp));
(void)closedir(dir);

/* Truncate any file name. */
*bup = '\0';
}

notfound:
/*
 * If readdir set errno, use it, not any saved error; otherwise,
 * didn't find the current directory in its parent directory, set
 * errno to ENOENT.
 */
if (!errno)
    errno = save_errno ? save_errno : ENOENT;
/* FALLTHROUGH */
err:
if (ptsize)
    free(pt);
if (up)
    free(up);
if (dir)
    (void)closedir(dir);
return NULL;
}
#endif /* HAVE_GETCWD */
/*
 * Copyright (c) 2008-2014 Todd C. Miller <Todd.Miller@courtesan.com>
 * Copyright (c) 1989, 1993
 * The Regents of the University of California. All rights reserved.
 *
 * This code is derived from software contributed to Berkeley by
 * Guido van Rossum.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:

```

```

* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. Neither the name of the University nor the names of its contributors
*   may be used to endorse or promote products derived from this software
*   without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* @(#)glob.c8.3 (Berkeley) 10/13/93
*/

/*
* glob(3) -- a superset of the one defined in POSIX 1003.2.
*
* The [!...] convention to negate a range is supported (SysV, Posix, ksh).
*
* Optional extra services, controlled by flags not defined by POSIX:
*
* GLOB_MAGCHAR:
* Set in gl_flags if pattern contained a globbing character.
* GLOB_TILDE:
* expand ~user/foo to the /home/dir/of/user/foo
* GLOB_BRACE:
* expand {1,2}{a,b} to 1a 1b 2a 2b
* gl_matchc:
* Number of matches in the current invocation of glob.
*/

#include <config.h>

#ifndef HAVE_GLOB

#include <sys/types.h>
#include <sys/stat.h>

#include <stdio.h>
#include <stdlib.h>
#ifdef HAVE_STRING_H
# include <string.h>
#endif /* HAVE_STRING_H */
#ifdef HAVE_STRINGS_H

```

```
# include <strings.h>
#endif /* HAVE_STRINGS_H */
#include <unistd.h>
#if defined(HAVE_STDINT_H)
# include <stdint.h>
#elif defined(HAVE_INTTYPES_H)
# include <inttypes.h>
#endif
#include <ctype.h>
#include <dirent.h>
#include <errno.h>
#include <limits.h>
#include <pwd.h>

#include "sudo_compat.h"
#include "compat/glob.h"
#include "compat/charclass.h"

#defineDOLLAR'$'
#defineDOT'.'
#defineEOS'\0'
#defineLBRACKET '['
#defineNOT'!'
#defineQUESTION'?'
#defineQUOTE'\''
#defineRANGE'-'
#defineRBRACKET']'
#defineSEP'/'
#defineSTAR'*'
#defineTILDE'~'
#defineUNDERSCORE'_'
#defineLBRACE'{'
#defineRBRACE'}'
#defineSLASH '/'
#defineCOMMA','

#ifndef DEBUG

#defineM_QUOTE0x8000
#defineM_PROTECT0x4000
#defineM_MASK0xffff
#defineM_ASCII0x00ff

typedef unsigned short Char;

#else

#defineM_QUOTE0x80
#defineM_PROTECT0x40
#defineM_MASK0xff
#defineM_ASCII0x7f

typedef char Char;

#endif
```



```

#define CHAR(c)((Char)((c)&M_ASCII))
#define META(c)((Char)((c)|M_QUOTE))
#define M_ALLMETA('*')
#define M_ENDMETA(']')
#define M_NOTMETA('!')
#define M_ONEMETA('?')
#define M_RNGMETA('-',)
#define M_SETMETA('[')
#define M_CLASSMETA(':',)
#define ismeta(c)(((c)&M_QUOTE) != 0)

#define GLOB_LIMIT_MALLOC 65536
#define GLOB_LIMIT_STAT 2048
#define GLOB_LIMIT_READDIR 16384

/* Limit of recursion during matching attempts. */
#define GLOB_LIMIT_RECUR 64

struct glob_lim {
    size_t glm_malloc;
    size_t glm_stat;
    size_t glm_readdir;
};

static int compare(const void *, const void *);
static int g_Ctoc(const Char *, char *, unsigned int);
static int g_lstat(Char *, struct stat *, glob_t *);
static DIR *g_opendir(Char *, glob_t *);
static Char *g_strchr(const Char *, int);
static int g_strncmp(const Char *, const char *, size_t);
static int g_stat(Char *, struct stat *, glob_t *);
static int glob0(const Char *, glob_t *, struct glob_lim *);
static int glob1(Char *, Char *, glob_t *, struct glob_lim *);
static int glob2(Char *, Char *, Char *, Char *, Char *, Char *,
    glob_t *, struct glob_lim *);
static int glob3(Char *, Char *, Char *, Char *, Char *,
    Char *, Char *, glob_t *, struct glob_lim *);
static int globextend(const Char *, glob_t *, struct glob_lim *,
    struct stat *);
static const Char *
    globtilde(const Char *, Char *, size_t, glob_t *);
static int globexp1(const Char *, glob_t *, struct glob_lim *);
static int globexp2(const Char *, const Char *, glob_t *,
    struct glob_lim *);
static int match(Char *, Char *, Char *, int);
#ifdef DEBUG
static void qprintf(const char *, Char *);
#endif

int
sudo_glob(const char *pattern, int flags, int (*errfunc)(const char *, int),
    glob_t *pglob)

```

```

{
const unsigned char *patnext;
int c;
Char *bufnext, *bufend, patbuf[PATH_MAX];
struct glob_lim limit = { 0, 0, 0 };

patnext = (unsigned char *) pattern;
if (!(flags & GLOB_APPEND)) {
pglob->gl_pathc = 0;
pglob->gl_pathv = NULL;
if (!(flags & GLOB_DOOFFS))
pglob->gl_offs = 0;
}
pglob->gl_flags = flags & ~GLOB_MAGCHAR;
pglob->gl_errfunc = errfunc;
pglob->gl_matchc = 0;

if (pglob->gl_offs < 0 || pglob->gl_pathc < 0 ||
    pglob->gl_offs >= INT_MAX || pglob->gl_pathc >= INT_MAX ||
    pglob->gl_pathc >= INT_MAX - pglob->gl_offs - 1)
return GLOB_NOSPACE;

if (strlen(pattern, PATH_MAX) == PATH_MAX)
return GLOB_NOMATCH;

bufnext = patbuf;
bufend = bufnext + PATH_MAX - 1;
if (flags & GLOB_NOESCAPE)
while (bufnext < bufend && (c = *patnext++) != EOS)
*bufnext++ = c;
else {
/* Protect the quoted characters. */
while (bufnext < bufend && (c = *patnext++) != EOS)
if (c == QUOTE) {
if ((c = *patnext++) == EOS) {
c = QUOTE;
--patnext;
}
*bufnext++ = c | M_PROTECT;
} else
*bufnext++ = c;
}
*bufnext = EOS;

if (flags & GLOB_BRACE)
return globexp1(patbuf, pglob, &limit);
else
return glob0(patbuf, pglob, &limit);
}

/*
* Expand recursively a glob {} pattern. When there is no more expansion
* invoke the standard globbing routine to glob the rest of the magic
* characters
*/

```

```

static int
globexp1(const Char *pattern, glob_t *pglob, struct glob_lim *limitp)
{
    const Char* ptr = pattern;

    /* Protect a single {}, for find(1), like csh */
    if (pattern[0] == LBRACE && pattern[1] == RBRACE && pattern[2] == EOS)
        return glob0(pattern, pglob, limitp);

    if ((ptr = (const Char *) g_strchr(ptr, LBRACE)) != NULL)
        return globexp2(ptr, pattern, pglob, limitp);

    return glob0(pattern, pglob, limitp);
}

/*
 * Recursive brace globbing helper. Tries to expand a single brace.
 * If it succeeds then it invokes globexp1 with the new pattern.
 * If it fails then it tries to glob the rest of the pattern and returns.
 */
static int
globexp2(const Char *ptr, const Char *pattern, glob_t *pglob,
        struct glob_lim *limitp)
{
    int i, rv;
    Char *lm, *ls;
    const Char *pe, *pm, *pl;
    Char patbuf[PATH_MAX];

    /* copy part up to the brace */
    for (lm = patbuf, pm = pattern; pm != ptr; *lm++ = *pm++)
        continue;
    *lm = EOS;
    ls = lm;

    /* Find the balanced brace */
    for (i = 0, pe = ++ptr; *pe; pe++)
        if (*pe == LBRACKET) {
            /* Ignore everything between [] */
            for (pm = pe++; *pe != RBRACKET && *pe != EOS; pe++)
                continue;
            if (*pe == EOS) {
                /*
                 * We could not find a matching RBRACKET.
                 * Ignore and just look for RBRACE
                 */
                pe = pm;
            }
            } else if (*pe == LBRACE)
                i++;
            else if (*pe == RBRACE) {
                if (i == 0)
                    break;
            }
    }

```

```

i--;
}

/* Non matching braces; just glob the pattern */
if (i != 0 || *pe == EOS)
return glob0(patbuf, pglob, limitp);

for (i = 0, pl = pm = ptr; pm <= pe; pm++) {
switch (*pm) {
case LBRACKET:
/* Ignore everything between [] */
for (pl = pm++; *pm != RBRACKET && *pm != EOS; pm++)
continue;
if (*pm == EOS) {
/*
* We could not find a matching RBRACKET.
* Ignore and just look for RBACE
*/
pm = pl;
}
break;

case LBRACE:
i++;
break;

case RBACE:
if (i) {
i--;
break;
}
/* FALLTHROUGH */
case COMMA:
if (i && *pm == COMMA)
break;
else {
/* Append the current string */
for (lm = ls; (pl < pm); *lm++ = *pl++)
continue;

/*
* Append the rest of the pattern after the
* closing brace
*/
for (pl = pe + 1; (*lm++ = *pl++) != EOS; )
continue;

/* Expand the current pattern */
#ifdef DEBUG
qprintf("globexp2:", patbuf);
#endif
rv = globexp1(patbuf, pglob, limitp);
if (rv && rv != GLOB_NOMATCH)
return rv;
}
}
}

```

```
/* move after the comma, to the next string */
pl = pm + 1;
}
break;

default:
break;
}
}
return 0;
}

/*
 * expand tilde from the passwd file.
 */
static const Char *
globtilde(const Char *pattern, Char *patbuf, size_t patbuf_len, glob_t *pglob)
{
    struct passwd *pwd;
    char *h;
    const Char *p;
    Char *b, *eb;

    if (*pattern != TILDE || !(pglob->gl_flags & GLOB_TILDE))
        return pattern;

    /* Copy up to the end of the string or / */
    eb = &patbuf[patbuf_len - 1];
    for (p = pattern + 1, h = (char *) patbuf;
         h < (char *)eb && *p && *p != SLASH; *h++ = *p++)
        continue;

    *h = EOS;

    if (((char *) patbuf)[0] == EOS) {
        /*
         * handle a plain ~ or ~/ by expanding $HOME
         * first and then trying the password file
         */
        if ((h = getenv("HOME")) == NULL) {
            if ((pwd = getpwuid(getuid())) == NULL)
                return pattern;
            else
                h = pwd->pw_dir;
        }
        } else {
        /*
         * Expand a ~user
         */
        if ((pwd = getpwnam((char*) patbuf)) == NULL)
            return pattern;
        else
    
```

```

h = pwd->pw_dir;
}

/* Copy the home directory */
for (b = patbuf; b < eb && *h; *b++ = *h++)
continue;

/* Append the rest of the pattern */
while (b < eb && (*b++ = *p++) != EOS)
continue;
*b = EOS;

return patbuf;
}

static int
g_strncmp(const Char *s1, const char *s2, size_t n)
{
int rv = 0;

while (n--) {
rv = *(Char *)s1 - *(const unsigned char *)s2++;
if (rv)
break;
if (*s1++ == '\0')
break;
}
return rv;
}

static int
g_charclass(const Char **patternp, Char **bufnextp)
{
const Char *pattern = *patternp + 1;
Char *bufnext = *bufnextp;
const Char *colon;
struct cclass *cc;
size_t len;

if ((colon = g_strchr(pattern, ':')) == NULL || colon[1] != ']')
return 1; /* not a character class */

len = (size_t)(colon - pattern);
for (cc = cclasses; cc->name != NULL; cc++) {
if (!g_strncmp(pattern, cc->name, len) && cc->name[len] == '\0')
break;
}
if (cc->name == NULL)
return -1; /* invalid character class */
*bufnext++ = M_CLASS;
*bufnext++ = (Char)(cc - &cclasses[0]);
*bufnextp = bufnext;
*patternp += len + 3;

return 0;
}

```

```

}

/*
 * The main glob() routine: compiles the pattern (optionally processing
 * quotes), calls glob1() to do the real pattern matching, and finally
 * sorts the list (unless unsorted operation is requested). Returns 0
 * if things went well, nonzero if errors occurred. It is not an error
 * to find no matches.
 */
static int
glob0(const Char *pattern, glob_t *pglob, struct glob_lim *limitp)
{
    const Char *qpatnext;
    int c, err, oldpathc;
    Char *bufnext, patbuf[PATH_MAX];

    qpatnext = globtilde(pattern, patbuf, PATH_MAX, pglob);
    oldpathc = pglob->gl_pathc;
    bufnext = patbuf;

    /* We don't need to check for buffer overflow any more. */
    while ((c = *qpatnext++) != EOS) {
        switch (c) {
        case LBRACKET:
            c = *qpatnext;
            if (c == NOT)
                ++qpatnext;
            if (*qpatnext == EOS ||
                g_strchr(qpatnext+1, RBRACKET) == NULL) {
                *bufnext++ = LBRACKET;
                if (c == NOT)
                    --qpatnext;
                break;
            }
            *bufnext++ = M_SET;
            if (c == NOT)
                *bufnext++ = M_NOT;
            c = *qpatnext++;
            do {
                if (c == LBRACKET && *qpatnext == ':') {
                    do {
                        err = g_charclass(&qpatnext,
                            &bufnext);
                        if (err)
                            break;
                        c = *qpatnext++;
                    } while (c == LBRACKET && *qpatnext == ':');
                    if (err == -1 &&
                        !(pglob->gl_flags & GLOB_NOCHECK))
                        return GLOB_NOMATCH;
                    if (c == RBRACKET)
                        break;
                }
                *bufnext++ = CHAR(c);
            }

```

```

    if (*qpatnext == RANGE &&
        (c = qpatnext[1]) != RBRACKET) {
        *bufnext++ = M_RNG;
        *bufnext++ = CHAR(c);
        qpatnext += 2;
    }
    while ((c = *qpatnext++) != RBRACKET);
    pglob->gl_flags |= GLOB_MAGCHAR;
    *bufnext++ = M_END;
    break;
case QUESTION:
    pglob->gl_flags |= GLOB_MAGCHAR;
    *bufnext++ = M_ONE;
    break;
case STAR:
    pglob->gl_flags |= GLOB_MAGCHAR;
    /* collapse adjacent stars to one,
     * to avoid exponential behavior
     */
    if (bufnext == patbuf || bufnext[-1] != M_ALL)
        *bufnext++ = M_ALL;
    break;
default:
    *bufnext++ = CHAR(c);
    break;
}
}
*bufnext = EOS;
#ifdef DEBUG
qprintf("glob0:", patbuf);
#endif

if ((err = glob1(patbuf, patbuf + PATH_MAX - 1, pglob, limitp)) != 0)
    return err;

/*
 * If there was no match we are going to append the pattern
 * if GLOB_NOCHECK was specified.
 */
if (pglob->gl_pathc == oldpathc) {
    if ((pglob->gl_flags & GLOB_NOCHECK))
        return globextend(pattern, pglob, limitp, NULL);
    else
        return GLOB_NOMATCH;
}
if (!(pglob->gl_flags & GLOB_NOSORT)) {
    qsort(pglob->gl_pathv + pglob->gl_offs + oldpathc,
        pglob->gl_pathc - oldpathc, sizeof(char *), compare);
}
return 0;
}

static int
compare(const void *p, const void *q)
{

```



```

return strcmp(*(char **)p, *(char **)q);
}

static int
glob1(Char *pattern, Char *pattern_last, glob_t *pglob, struct glob_lim *limitp)
{
    Char pathbuf[PATH_MAX];

    /* A null pathname is invalid -- POSIX 1003.1 sect. 2.4. */
    if (*pattern == EOS)
        return 0;
    return glob2(pathbuf, pathbuf + PATH_MAX - 1,
        pathbuf, pathbuf + PATH_MAX - 1,
        pattern, pattern_last, pglob, limitp);
}

/*
 * The functions glob2 and glob3 are mutually recursive; there is one level
 * of recursion for each segment in the pattern that contains one or more
 * meta characters.
 */
static int
glob2(Char *pathbuf, Char *pathbuf_last, Char *pathend, Char *pathend_last,
    Char *pattern, Char *pattern_last, glob_t *pglob, struct glob_lim *limitp)
{
    struct stat sb;
    Char *p, *q;
    int anymeta;

    /*
     * Loop over pattern segments until end of pattern or until
     * segment with meta character found.
     */
    for (anymeta = 0;;) {
        if (*pattern == EOS) { /* End of pattern? */
            *pathend = EOS;

            if ((pglob->gl_flags & GLOB_LIMIT) &&
                limitp->glim_stat++ >= GLOB_LIMIT_STAT) {
                errno = 0;
                *pathend++ = SEP;
                *pathend = EOS;
                return GLOB_NOSPACE;
            }
            if (g_lstat(pathbuf, &sb, pglob))
                return 0;

            if (((pglob->gl_flags & GLOB_MARK) &&
                pathend[-1] != SEP) && (S_ISDIR(sb.st_mode) ||
                (S_ISLNK(sb.st_mode) &&
                (g_stat(pathbuf, &sb, pglob) == 0) &&
                S_ISDIR(sb.st_mode)))) {
                if (pathend+1 > pathend_last)
                    return 1;
            }
        }
    }

```

```

*pathend++ = SEP;
*pathend = EOS;
}
++pglob->gl_matchc;
return globextend(pathbuf, pglob, limitp, &sb);
}

/* Find end of next segment, copy tentatively to pathend. */
q = pathend;
p = pattern;
while (*p != EOS && *p != SEP) {
if (ismeta(*p))
anymeta = 1;
if (q+1 > pathend_last)
return 1;
*q++ = *p++;
}

if (!anymeta) { /* No expansion, do next segment. */
pathend = q;
pattern = p;
while (*pattern == SEP) {
if (pathend+1 > pathend_last)
return 1;
*pathend++ = *pattern++;
}
} else
/* Need expansion, recurse. */
return glob3(pathbuf, pathbuf_last, pathend,
pathend_last, pattern, p, pattern_last,
pglob, limitp);
}
/* NOTREACHED */
}

static int
glob3(Char *pathbuf, Char *pathbuf_last, Char *pathend, Char *pathend_last,
Char *pattern, Char *restpattern, Char *restpattern_last, glob_t *pglob,
struct glob_lim *limitp)
{
struct dirent *dp;
DIR *dirp;
int err;
char buf[PATH_MAX];

if (pathend > pathend_last)
return 1;
*pathend = EOS;
errno = 0;

if ((dirp = g_opendir(pathbuf, pglob)) == NULL) {
/* TODO: don't call for ENOENT or ENOTDIR? */
if (pglob->gl_errfunc) {
if (g_Ctoc(pathbuf, buf, sizeof(buf)))
return GLOB_ABORTED;
}
}

```

```

    if (pglob->gl_errfunc(buf, errno) ||
        pglob->gl_flags & GLOB_ERR)
    return GLOB_ABORTED;
}
return 0;
}

err = 0;

/* Search directory for matching names. */
while ((dp = readdir(dirp))) {
    unsigned char *sc;
    Char *dc;

    if ((pglob->gl_flags & GLOB_LIMIT) &&
        limitp->glim_readdir++ >= GLOB_LIMIT_READDIR) {
        errno = 0;
        *pathend++ = SEP;
        *pathend = EOS;
        err = GLOB_NOSPACE;
        break;
    }

    /* Initial DOT must be matched literally. */
    if (dp->d_name[0] == DOT && *pattern != DOT)
        continue;
    dc = pathend;
    sc = (unsigned char *) dp->d_name;
    while (dc < pathend_last && (*dc++ = *sc++) != EOS)
        continue;
    if (dc >= pathend_last) {
        *dc = EOS;
        err = 1;
        break;
    }

    if (!match(pathend, pattern, restpattern, GLOB_LIMIT_RECUR)) {
        *pathend = EOS;
        continue;
    }
    err = glob2(pathbuf, pathbuf_last, --dc, pathend_last,
        restpattern, restpattern_last, pglob, limitp);
    if (err)
        break;
}

closedir(dirp);
return err;
}

/*
 * Extend the gl_pathv member of a glob_t structure to accommodate a new item,
 * add the new item, and update gl_pathc.
 */

```

```

* This assumes the BSD realloc, which only copies the block when its size
* crosses a power-of-two boundary; for v7 realloc, this would cause quadratic
* behavior.
*
* Return 0 if new item added, error code if memory couldn't be allocated.
*
* Invariant of the glob_t structure:
* Either gl_pathc is zero and gl_pathv is NULL; or gl_pathc > 0 and
* gl_pathv points to (gl_offs + gl_pathc + 1) items.
*/
static int
globextend(const Char *path, glob_t *pglob, struct glob_lim *limitp,
           struct stat *sb)
{
    char **pathv;
    ssize_t i;
    size_t newn, len;
    char *copy = NULL;
    const Char *p;

    newn = 2 + pglob->gl_pathc + pglob->gl_offs;
    if (pglob->gl_offs >= INT_MAX ||
        pglob->gl_pathc >= INT_MAX ||
        newn >= INT_MAX ||
        SIZE_MAX / sizeof(*pathv) <= newn) {
        nospace:
        for (i = pglob->gl_offs; i < (ssize_t)(newn - 2); i++) {
            if (pglob->gl_pathv && pglob->gl_pathv[i])
                free(pglob->gl_pathv[i]);
        }
        if (pglob->gl_pathv) {
            free(pglob->gl_pathv);
            pglob->gl_pathv = NULL;
        }
        return GLOB_NOSPACE;
    }

    pathv = reallocarray(pglob->gl_pathv, newn, sizeof(*pathv));
    if (pathv == NULL)
        goto nospace;
    if (pglob->gl_pathv == NULL && pglob->gl_offs > 0) {
        /* first time around -- clear initial gl_offs items */
        pathv += pglob->gl_offs;
        for (i = pglob->gl_offs; --i >= 0; )
            *--pathv = NULL;
    }
    pglob->gl_pathv = pathv;

    for (p = path; *p++;)
        continue;
    len = (size_t)(p - path);
    limitp->glim_malloc += len;
    if ((copy = malloc(len)) != NULL) {
        if (g_Ctoc(path, copy, len)) {
            free(copy);

```

```

return GLOB_NOSPACE;
}
pathv[pglob->gl_offs + pglob->gl_pathc++] = copy;
}
pathv[pglob->gl_offs + pglob->gl_pathc] = NULL;

if ((pglob->gl_flags & GLOB_LIMIT) &&
    (newn * sizeof(*pathv)) + limitp->glim_malloc >
    GLOB_LIMIT_MALLOC) {
    errno = 0;
    return GLOB_NOSPACE;
}
return copy == NULL ? GLOB_NOSPACE : 0;
}

/*
 * pattern matching function for filenames. Each occurrence of the *
 * pattern causes a recursion level.
 */
static int
match(Char *name, Char *pat, Char *patend, int recur)
{
    int ok, negate_range;
    Char c, k;

    if (recur-- == 0)
        return GLOB_NOSPACE;

    while (pat < patend) {
        c = *pat++;
        switch (c & M_MASK) {
        case M_ALL:
            while (pat < patend && (*pat & M_MASK) == M_ALL)
                pat++; /* eat consecutive '*' */
            if (pat == patend)
                return 1;
            do {
                if (match(name, pat, patend, recur))
                    return 1;
            } while (*name++ != EOS);
            return 0;
        case M_ONE:
            if (*name++ == EOS)
                return 0;
            break;
        case M_SET:
            ok = 0;
            if ((k = *name++) == EOS)
                return 0;
            if ((negate_range = ((*pat & M_MASK) == M_NOT)) != EOS)
                ++pat;
            while (((c = *pat++) & M_MASK) != M_END) {
                if ((c & M_MASK) == M_CLASS) {

```

```

Char idx = *pat & M_MASK;
if (idx < NCCLASSES &&
    cclasses[idx].isctype(k))
ok = 1;
++pat;
}
if ((*pat & M_MASK) == M_RNG) {
if (c <= k && k <= pat[1])
ok = 1;
pat += 2;
} else if (c == k)
ok = 1;
}
if (ok == negate_range)
return 0;
break;
default:
if (*name++ != c)
return 0;
break;
}
}
return *name == EOS;
}

/* Free allocated data belonging to a glob_t structure. */
void
sudo_globfree(glob_t *pglob)
{
int i;
char **pp;

if (pglob->gl_pathv != NULL) {
pp = pglob->gl_pathv + pglob->gl_offs;
for (i = pglob->gl_pathc; i--; ++pp)
if (*pp)
free(*pp);
free(pglob->gl_pathv);
pglob->gl_pathv = NULL;
}
}

static DIR *
g_opendir(Char *str, glob_t *pglob)
{
char buf[PATH_MAX];

if (!*str) {
buf[0] = '.';
buf[1] = '\0';
} else {
if (g_Ctoc(str, buf, sizeof(buf)))
return NULL;
}
}

```

```
return opendir(buf);
}

static int
g_lstat(Char *fn, struct stat *sb, glob_t *pglob)
{
char buf[PATH_MAX];

if (g_Ctoc(fn, buf, sizeof(buf)))
return -1;
return lstat(buf, sb);
}

static int
g_stat(Char *fn, struct stat *sb, glob_t *pglob)
{
char buf[PATH_MAX];

if (g_Ctoc(fn, buf, sizeof(buf)))
return -1;
return stat(buf, sb);
}

static Char *
g_strchr(const Char *str, int ch)
{
do {
if (*str == ch)
return (Char *)str;
} while (*str++);
return NULL;
}

static int
g_Ctoc(const Char *str, char *buf, unsigned int len)
{
while (len--) {
if ((*buf++ = *str++) == EOS)
return 0;
}
return 1;
}

#ifdef DEBUG
static void
qprintf(const char *str, Char *s)
{
Char *p;

(void)printf("%s:\n", str);
for (p = s; *p; p++)
(void)printf("%c", CHAR(*p));
(void)printf("\n");
}
```

```

for (p = s; *p; p++)
(void)printf("%c", *p & M_PROTECT ? "" : '');
(void)printf("\n");
for (p = s; *p; p++)
(void)printf("%c", ismeta(*p) ? ' _ ' : '');
(void)printf("\n");
}
#endif /* DEBUG */
#endif /* HAVE_GLOB */
/*$OpenBSD: inet_pton.c,v 1.8 2010/05/06 15:47:14 claudio Exp $*/

/* Copyright (c) 1996 by Internet Software Consortium.
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
 * ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
 * CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
 * DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
 * PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
 * ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
 * SOFTWARE.
 */

#include <config.h>

#if !defined(HAVE_INET_PTON)

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <arpa/nameser.h>
#ifdef HAVE_STRING_H
# include <string.h>
#endif /* HAVE_STRING_H */
#ifdef HAVE_STRINGS_H
# include <strings.h>
#endif /* HAVE_STRINGS_H */
#include <errno.h>

#include "sudo_compat.h"

#ifdef EAFNOSUPPORT
# define EAFNOSUPPORT EINVAL
#endif

#ifdef NS_INADDRSZ
# ifdef INADDRSZ
# define NS_INADDRSZ INADDRSZ
# else
# define NS_INADDRSZ 4

```



```
# endif
#endif
#ifndef NS_IN6ADDRSZ
# ifdef IN6ADDRSZ
#  define NS_IN6ADDRSZ IN6ADDRSZ
# else
#  define NS_IN6ADDRSZ 16
# endif
#endif
#ifndef NS_INT16SZ
# ifdef INT16SZ
#  define NS_INT16SZ INT16SZ
# else
#  define NS_INT16SZ 2
# endif
#endif

/*
 * WARNING: Don't even consider trying to compile this on a system where
 * sizeof(int) < 4.  sizeof(int) > 4 is fine; all the world's not a VAX.
 */

/* int
 * inet_pton4(src, dst)
 *like inet_aton() but without all the hexadecimal and shorthand.
 * return:
 * 1 if `src' is a valid dotted quad, else 0.
 * notice:
 * does not touch `dst' unless it's returning 1.
 * author:
 * Paul Vixie, 1996.
 */
static int
inet_pton4(const char *src, u_char *dst)
{
    const char digits[] = "0123456789";
    int saw_digit, octets, ch;
    u_char tmp[NS_INADDRSZ], *tp;

    saw_digit = 0;
    octets = 0;
    /* cppcheck-suppress uninitvar */
    *(tp = tmp) = '\0';
    while ((ch = (unsigned char)*src++) != '\0') {
        const char *pch;

        if ((pch = strchr(digits, ch)) != NULL) {
            u_int new = *tp * 10 + (pch - digits);

            if (new > 255)
                return (0);
            if (!saw_digit) {
                if (++octets > 4)
                    return (0);
            }
            *tp++ = (u_char)new;
        }
    }
    *tp = '\0';
    return (octets == 4);
}
```

```

    saw_digit = 1;
    }
    *tp = new;
    } else if (ch == '.' && saw_digit) {
    if (octets == 4)
    return (0);
    *++tp = 0;
    saw_digit = 0;
    } else
    return (0);
    }
    if (octets < 4)
    return (0);

    memcpy(dst, tmp, NS_INADDRSZ);
    return (1);
    }

#ifdef HAVE_STRUCT_IN6_ADDR
/* int
 * inet_pton6(src, dst)
 * convert presentation level address to network order binary form.
 * return:
 * 1 if `src' is a valid [RFC1884 2.2] address, else 0.
 * notice:
 * does not touch `dst' unless it's returning 1.
 * credit:
 * inspired by Mark Andrews.
 * author:
 * Paul Vixie, 1996.
 */
static int
inet_pton6(const char *src, u_char *dst)
{
    const char xdigits_l[] = "0123456789abcdef",
        xdigits_u[] = "0123456789ABCDEF";
    u_char tmp[NS_IN6ADDRSZ], *tp, *endp, *colonp;
    const char *xdigits, *curtok;
    int ch, saw_xdigit, count_xdigit;
    u_int val;

    /* cppcheck-suppress uninitvar */
    memset((tp = tmp), 0, NS_IN6ADDRSZ);
    endp = tp + NS_IN6ADDRSZ;
    colonp = NULL;
    /* Leading :: requires some special handling. */
    if (*src == ':')
    if (*++src != ':')
    return (0);
    curtok = src;
    saw_xdigit = count_xdigit = 0;
    val = 0;
    while ((ch = (unsigned char)*src++) != '\0') {
        const char *pch;

```

```

if ((pch = strchr((xdigits = xdigits_l), ch)) == NULL)
pch = strchr((xdigits = xdigits_u), ch);
if (pch != NULL) {
if (count_xdigit >= 4)
return (0);
val <<= 4;
val |= (pch - xdigits);
if (val > 0xffff)
return (0);
saw_xdigit = 1;
count_xdigit++;
continue;
}
if (ch == ':') {
curtok = src;
if (!saw_xdigit) {
if (colonp)
return (0);
colonp = tp;
continue;
} else if (*src == '\0') {
return (0);
}
if (tp + NS_INT16SZ > endp)
return (0);
*tp++ = (u_char) (val >> 8) & 0xff;
*tp++ = (u_char) val & 0xff;
saw_xdigit = 0;
count_xdigit = 0;
val = 0;
continue;
}
if (ch == '.' && ((tp + NS_INADDRSZ) <= endp) &&
    inet_pton4(curtok, tp) > 0) {
tp += NS_INADDRSZ;
saw_xdigit = 0;
count_xdigit = 0;
break; /* '\0' was seen by inet_pton4(). */
}
return (0);
}
if (saw_xdigit) {
if (tp + NS_INT16SZ > endp)
return (0);
*tp++ = (u_char) (val >> 8) & 0xff;
*tp++ = (u_char) val & 0xff;
}
if (colonp != NULL) {
/*
 * Since some memmove()'s erroneously fail to handle
 * overlapping regions, we'll do the shift by hand.
 */
const long n = tp - colonp;
long i;

```

```

if (tp == endp)
return (0);
for (i = 1; i <= n; i++) {
endp[- i] = colonp[n - i];
colonp[n - i] = 0;
}
tp = endp;
}
if (tp != endp)
return (0);
memcpy(dst, tmp, NS_IN6ADDRSZ);
return (1);
}
#endif /* HAVE_STRUCT_IN6_ADDR */

/* int
 * inet_pton(af, src, dst)
 *convert from presentation format (which usually means ASCII printable)
 *to network format (which is usually some kind of binary format).
 * return:
 * 1 if the address was valid for the specified address family
 * 0 if the address wasn't valid ('dst' is untouched in this case)
 *-1 if some other error occurred ('dst' is untouched in this case, too)
 * author:
 *Paul Vixie, 1996.
 */
int
sudo_inet_pton(int af, const char *src, void *dst)
{
switch (af) {
case AF_INET:
return (inet_pton4(src, dst));
#ifdef HAVE_STRUCT_IN6_ADDR
case AF_INET6:
return (inet_pton6(src, dst));
#endif /* HAVE_STRUCT_IN6_ADDR */
default:
errno = EAFNOSUPPORT;
return (-1);
}
}
/* NOTREACHED */
}

```

#endif /* HAVE_INET_PTON */

Sudo is distributed under the following license:

Copyright (c) 1994-1996, 1998-2015

Todd C. Miller <Todd.Miller@courtesan.com>

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES

WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F39502-99-1-0512.

The file `redblack.c` bears the following license:

Copyright (c) 2001 Emin Martinian

Redistribution and use in source and binary forms, with or without modification, are permitted provided that neither the name of Emin Martinian nor the names of any contributors are be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The file `reallocarray.c` bears the following license:

Copyright (c) 2008 Otto Moerbeek <otto@drijf.net>

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The files `getcwd.c`, `glob.c`, `glob.h`, `snprintf.c` and `sudo_queue.h` bear the following license:

Copyright (c) 1989, 1990, 1991, 1993

The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The file fnmatch.c bears the following license:

Copyright (c) 2011, VMware, Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the VMware, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL VMWARE, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The file getopt_long.c bears the following license:

/*_

```
* Copyright (c) 2000 The NetBSD Foundation, Inc.
* All rights reserved.
*
* This code is derived from software contributed to The NetBSD Foundation
* by Dieter Baron and Thomas Klausner.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
*
* THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS
* ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
* TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS
* BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
* SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*/
```

The file inet_pton.c bears the following license:

```
/* Copyright (c) 1996 by Internet Software Consortium.
*
* Permission to use, copy, modify, and distribute this software for any
* purpose with or without fee is hereby granted, provided that the above
* copyright notice and this permission notice appear in all copies.
*
* THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS
* ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE
* CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL
* DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
* PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
* ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
* SOFTWARE.
*/
```

The embedded copy of zlib bears the following license:

Copyright (C) 1995-2013 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly Mark Adler

jlopou@gzip.org madler@alumni.caltech.edu

/*\$OpenBSD: reallocarray.c,v 1.2 2014/12/08 03:45:00 bcook Exp \$*/

/*

* Copyright (c) 2008 Otto Moerbeek <otto@drijf.net>

*

* Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.

*

* THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
 * WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
 * MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
 * ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
 * WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
 * ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
 * OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

*/

#include <config.h>

#ifndef HAVE_REALLOCARRAY

#include <sys/types.h>

#include <stdlib.h>

#if defined(HAVE_STDINT_H)

include <stdint.h>

#elif defined(HAVE_INTTYPES_H)

include <inttypes.h>

#endif

#include <errno.h>

#include <limits.h>

#include "sudo_compat.h"

/*

* This is sqrt(SIZE_MAX+1), as s1*s2 <= SIZE_MAX

* if both s1 < MUL_NO_OVERFLOW and s2 < MUL_NO_OVERFLOW

*/

#define MUL_NO_OVERFLOW((size_t)1 << (sizeof(size_t) * 4))

void *

sudo_reallocarray(void *optr, size_t nmemb, size_t size)


```
{
if ((nmemb >= MUL_NO_OVERFLOW || size >= MUL_NO_OVERFLOW) &&
    nmemb > 0 && SIZE_MAX / nmemb < size) {
errno = ENOMEM;
return NULL;
}
return realloc(optr, size * nmemb);
}

#endif /* HAVE_REALLOCARRAY */
/*
 * Copyright (c) 2004-2005, 2007, 2009-2015
 * Todd C. Miller <Todd.Miller@courtesan.com>
 *
 * Permission to use, copy, modify, and distribute this software for any
 * purpose with or without fee is hereby granted, provided that the above
 * copyright notice and this permission notice appear in all copies.
 *
 * THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
 * WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
 * MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
 * ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
 * WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
 * ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
 * OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
 */

/*
 * Adapted from the following code written by Emin Martinian:
 * http://web.mit.edu/~emin/www/source_code/red_black_tree/index.html
 *
 * Copyright (c) 2001 Emin Martinian
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that neither the name of Emin
 * Martinian nor the names of any contributors are be used to endorse or
 * promote products derived from this software without specific prior
 * written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#include <config.h>
```

```

#include <sys/types.h>

#include <stdio.h>
#include <stdlib.h>

#include "sudoers.h"
#include "redblack.h"

static void rbrepair(struct rbtree *, struct rbnode *);
static void rotate_left(struct rbtree *, struct rbnode *);
static void rotate_right(struct rbtree *, struct rbnode *);
static void rbdestroy_int(struct rbtree *, struct rbnode *, void (*)(void *));

/*
 * Red-Black tree, see http://en.wikipedia.org/wiki/Red-black\_tree
 *
 * A red-black tree is a binary search tree where each node has a color
 * attribute, the value of which is either red or black. Essentially, it
 * is just a convenient way to express a 2-3-4 binary search tree where
 * the color indicates whether the node is part of a 3-node or a 4-node.
 * In addition to the ordinary requirements imposed on binary search
 * trees, we make the following additional requirements of any valid
 * red-black tree:
 * 1) Every node is either red or black.
 * 2) The root is black.
 * 3) All leaves are black.
 * 4) Both children of each red node are black.
 * 5) The paths from each leaf up to the root each contain the same
 *    number of black nodes.
 */

/*
 * Create a red black tree struct using the specified compare routine.
 * Allocates and returns the initialized (empty) tree or NULL if
 * memory cannot be allocated.
 */
struct rbtree *
rbcreate(int (*compar)(const void *, const void*))
{
    struct rbtree *tree;
    debug_decl(rbcreate, SUDOERS_DEBUG_RBTREE)

    if ((tree = malloc(sizeof(*tree))) == NULL) {
        sudo_debug_printf(SUDO_DEBUG_ERROR|SUDO_DEBUG_LINENO,
            "unable to allocate memory");
        debug_return_ptr(NULL);
    }

    tree->compar = compar;

    /*
     * We use a self-referencing sentinel node called nil to simplify the
     * code by avoiding the need to check for NULL pointers.
     */
    tree->nil.left = tree->nil.right = tree->nil.parent = &tree->nil;

```

```
tree->nil.color = black;
tree->nil.data = NULL;

/*
 * Similarly, the fake root node keeps us from having to worry
 * about splitting the root.
 */
tree->root.left = tree->root.right = tree->root.parent = &tree->nil;
tree->root.color = black;
tree->root.data = NULL;

debug_return_ptr(tree);
}

/*
 * Perform a left rotation starting at node.
 */
static void
rotate_left(struct rbtree *tree, struct rbnode *node)
{
    struct rbnode *child;
    debug_decl(rotate_left, SUDOERS_DEBUG_RBTREE)

    child = node->right;
    node->right = child->left;

    if (child->left != rbnil(tree))
        child->left->parent = node;
    child->parent = node->parent;

    if (node == node->parent->left)
        node->parent->left = child;
    else
        node->parent->right = child;
    child->left = node;
    node->parent = child;

    debug_return;
}

/*
 * Perform a right rotation starting at node.
 */
static void
rotate_right(struct rbtree *tree, struct rbnode *node)
{
    struct rbnode *child;
    debug_decl(rotate_right, SUDOERS_DEBUG_RBTREE)

    child = node->left;
    node->left = child->right;

    if (child->right != rbnil(tree))
        child->right->parent = node;
```

```

    child->parent = node->parent;

    if (node == node->parent->left)
node->parent->left = child;
    else
node->parent->right = child;
    child->right = node;
    node->parent = child;

    debug_return;
}

/*
 * Insert data pointer into a redblack tree.
 * Returns a 0 on success, 1 if a node matching "data" already exists
 * (filling in "existing" if not NULL), or -1 on malloc() failure.
 */
int
rbinsert(struct rbtree *tree, void *data, struct rbnode **existing)
{
    struct rbnode *node = rbfirst(tree);
    struct rbnode *parent = rbroot(tree);
    int res;
    debug_decl(rbinsert, SUDOERS_DEBUG_RBTREE)

    /* Find correct insertion point. */
    while (node != rbnil(tree)) {
parent = node;
        if ((res = tree->compar(data, node->data)) == 0) {
            if (existing != NULL)
                *existing = node;
            debug_return_int(1);
        }
        node = res < 0 ? node->left : node->right;
    }

    node = malloc(sizeof(*node));
    if (node == NULL) {
sudo_debug_printf(SUDO_DEBUG_ERROR|SUDO_DEBUG_LINENO,
        "unable to allocate memory");
        debug_return_int(-1);
    }
    node->data = data;
    node->left = node->right = rbnil(tree);
    node->parent = parent;
    if (parent == rbroot(tree) || tree->compar(data, parent->data) < 0)
parent->left = node;
    else
parent->right = node;
    node->color = red;

    /*
     * If the parent node is black we are all set, if it is red we have
     * the following possible cases to deal with. We iterate through
     * the rest of the tree to make sure none of the required properties

```

```

* is violated.
*
* 1) The uncle is red. We repaint both the parent and uncle black
*    and repaint the grandparent node red.
*
* 2) The uncle is black and the new node is the right child of its
*    parent, and the parent in turn is the left child of its parent.
*    We do a left rotation to switch the roles of the parent and
*    child, relying on further iterations to fixup the old parent.
*
* 3) The uncle is black and the new node is the left child of its
*    parent, and the parent in turn is the left child of its parent.
*    We switch the colors of the parent and grandparent and perform
*    a right rotation around the grandparent. This makes the former
*    parent the parent of the new node and the former grandparent.
*
* Note that because we use a sentinel for the root node we never
* need to worry about replacing the root.
*/
while (node->parent->color == red) {
struct rbnode *uncle;
if (node->parent == node->parent->parent->left) {
    uncle = node->parent->parent->right;
    if (uncle->color == red) {
node->parent->color = black;
uncle->color = black;
node->parent->parent->color = red;
node = node->parent->parent;
    } else /* if (uncle->color == black) */ {
if (node == node->parent->right) {
    node = node->parent;
    rotate_left(tree, node);
}
node->parent->color = black;
node->parent->parent->color = red;
rotate_right(tree, node->parent->parent);
}
} else /* if (node->parent == node->parent->parent->right) */
    uncle = node->parent->parent->left;
    if (uncle->color == red) {
node->parent->color = black;
uncle->color = black;
node->parent->parent->color = red;
node = node->parent->parent;
    } else /* if (uncle->color == black) */ {
if (node == node->parent->left) {
    node = node->parent;
    rotate_right(tree, node);
}
node->parent->color = black;
node->parent->parent->color = red;
rotate_left(tree, node->parent->parent);
}
}
}

```

```

    }
    rbfirst(tree)->color = black; /* first node is always black */
    debug_return_int(0);
}

/*
 * Look for a node matching key in tree.
 * Returns a pointer to the node if found, else NULL.
 */
struct rbnode *
rbfind(struct rbtree *tree, void *key)
{
    struct rbnode *node = rbfirst(tree);
    int res;
    debug_decl(rbfind, SUDOERS_DEBUG_RBTREE)

    while (node != rbnil(tree)) {
        if ((res = tree->compar(key, node->data)) == 0)
            debug_return_ptr(node);
        node = res < 0 ? node->left : node->right;
    }
    debug_return_ptr(NULL);
}

/*
 * Call func() for each node, passing it the node data and a cookie;
 * If func() returns non-zero for a node, the traversal stops and the
 * error value is returned. Returns 0 on successful traversal.
 */
int
rbapply_node(struct rbtree *tree, struct rbnode *node,
             int (*func)(void *, void *), void *cookie, enum rbtraversal order)
{
    int error;
    debug_decl(rbapply_node, SUDOERS_DEBUG_RBTREE)

    if (node != rbnil(tree)) {
        if (order == preorder)
            if ((error = func(node->data, cookie)) != 0)
                debug_return_int(error);
        if ((error = rbapply_node(tree, node->left, func, cookie, order)) != 0)
            debug_return_int(error);
        if (order == inorder)
            if ((error = func(node->data, cookie)) != 0)
                debug_return_int(error);
        if ((error = rbapply_node(tree, node->right, func, cookie, order)) != 0)
            debug_return_int(error);
        if (order == postorder)
            if ((error = func(node->data, cookie)) != 0)
                debug_return_int(error);
    }
    debug_return_int(0);
}

/*

```

```

    * Returns the successor of node, or nil if there is none.
    */
static struct rbnode *
rbsuccessor(struct rbtree *tree, struct rbnode *node)
{
    struct rbnode *succ;
    debug_decl(rbsuccessor, SUDOERS_DEBUG_RBTREE)

    if ((succ = node->right) != rbnil(tree)) {
while (succ->left != rbnil(tree))
        succ = succ->left;
    } else {
/* No right child, move up until we find it or hit the root */
for (succ = node->parent; node == succ->right; succ = succ->parent)
    node = succ;
if (succ == rbroot(tree))
    succ = rbnil(tree);
    }
    debug_return_ptr(succ);
}

/*
 * Recursive portion of rbdestroy().
 */
static void
rbdestroy_int(struct rbtree *tree, struct rbnode *node, void (*destroy)(void *))
{
    debug_decl(rbdestroy_int, SUDOERS_DEBUG_RBTREE)
    if (node != rbnil(tree)) {
rbdestroy_int(tree, node->left, destroy);
rbdestroy_int(tree, node->right, destroy);
if (destroy != NULL)
    destroy(node->data);
free(node);
    }
    debug_return;
}

/*
 * Destroy the specified tree, calling the destructor "destroy"
 * for each node and then freeing the tree itself.
 */
void
rbdestroy(struct rbtree *tree, void (*destroy)(void *))
{
    debug_decl(rbdestroy, SUDOERS_DEBUG_RBTREE)
    rbdestroy_int(tree, rbfirst(tree), destroy);
    free(tree);
    debug_return;
}

/*
 * Delete node 'z' from the tree and return its data pointer.
 */

```

```

void *rbdelete(struct rbtree *tree, struct rbnode *z)
{
    struct rbnode *x, *y;
    void *data = z->data;
    debug_decl(rbdelete, SUDOERS_DEBUG_RBTREE)

    if (z->left == rbnil(tree) || z->right == rbnil(tree))
        y = z;
    else
        y = rbsuccessor(tree, z);
    x = (y->left == rbnil(tree)) ? y->right : y->left;

    if ((x->parent = y->parent) == rbroot(tree)) {
        rbfirst(tree) = x;
    } else {
        if (y == y->parent->left)
            y->parent->left = x;
        else
            y->parent->right = x;
    }
    if (y->color == black)
        rbrepair(tree, x);
    if (y != z) {
        y->left = z->left;
        y->right = z->right;
        y->parent = z->parent;
        y->color = z->color;
        z->left->parent = z->right->parent = y;
        if (z == z->parent->left)
            z->parent->left = y;
        else
            z->parent->right = y;
    }
    free(z);

    debug_return_ptr(data);
}

/*
 * Repair the tree after a node has been deleted by rotating and repainting
 * colors to restore the 4 properties inherent in red-black trees.
 */
static void
rbrepair(struct rbtree *tree, struct rbnode *node)
{
    struct rbnode *sibling;
    debug_decl(rbrepair, SUDOERS_DEBUG_RBTREE)

    while (node->color == black && node != rbfirst(tree)) {
        if (node == node->parent->left) {
            sibling = node->parent->right;
            if (sibling->color == red) {
                sibling->color = black;
                node->parent->color = red;
                rotate_left(tree, node->parent);
            }
        }
    }

```



```

    sibling = node->parent->right;
    }
    if (sibling->right->color == black && sibling->left->color == black) {
sibling->color = red;
node = node->parent;
    } else {
if (sibling->right->color == black) {
    sibling->left->color = black;
    sibling->color = red;
    rotate_right(tree, sibling);
    sibling = node->parent->right;
}
sibling->color = node->parent->color;
node->parent->color = black;
sibling->right->color = black;
rotate_left(tree, node->parent);
node = rbfirst(tree); /* exit loop */
    }
} else { /* if (node == node->parent->right) */
    sibling = node->parent->left;
    if (sibling->color == red) {
sibling->color = black;
node->parent->color = red;
rotate_right(tree, node->parent);
sibling = node->parent->left;
    }
    if (sibling->right->color == black && sibling->left->color == black) {
sibling->color = red;
node = node->parent;
    } else {
if (sibling->left->color == black) {
    sibling->right->color = black;
    sibling->color = red;
    rotate_left(tree, sibling);
    sibling = node->parent->left;
}
sibling->color = node->parent->color;
node->parent->color = black;
sibling->left->color = black;
rotate_right(tree, node->parent);
node = rbfirst(tree); /* exit loop */
    }
}
}
    node->color = black;

    debug_return;
}
/*$OpenBSD: vfprintf.c,v 1.67 2014/12/21 00:23:30 daniel Exp $*/
/*_
 * Copyright (c) 1999-2005, 2008, 2010-2016
 *   Todd C. Miller <Todd.Miller@courtesan.com>
 * Copyright (c) 1990, 1993
 * The Regents of the University of California. All rights reserved.

```

```

*
* This code is derived from software contributed to Berkeley by
* Chris Torek.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. Neither the name of the University nor the names of its contributors
*   may be used to endorse or promote products derived from this software
*   without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* From: @(#)vfprintf.c 8.1 (Berkeley) 6/4/93
*/

/*
* vfprintf/vprintf based on OpenBSD vfprintf.c.
*/

#include <config.h>

#if !defined(HAVE_VSNPRINTF) || !defined(HAVE_SNPRINTF) || \
    !defined(HAVE_VASPRINTF) || !defined(HAVE_ASPRINTF) || \
    defined(PREFER_PORTABLE_SNPRINTF)

#include <sys/types.h>
#include <sys/mman.h>

#include <errno.h>
#ifdef HAVE_NL_LANGINFO
#include <langinfo.h>
#endif
#include <limits.h>
#include <stdarg.h>
#include <stdlib.h>
#include <stddef.h>
#ifdef HAVE_STDINT_H
#include <stdint.h>
#elif defined(HAVE_INTTYPES_H)

```

```

#include <inttypes.h>
#endif
#include <stdio.h>
#ifdef HAVE_STRING_H
#include <string.h>
#endif /* HAVE_STRING_H */
#ifdef HAVE_STRINGS_H
#include <strings.h>
#endif /* HAVE_STRINGS_H */
#include <unistd.h>
#ifdef PRINTF_WIDE_CHAR
#include <wchar.h>
#endif
#include <fcntl.h>

#include "sudo_compat.h"

/* Avoid printf format attacks by ignoring the %n escape. */
#define NO_PRINTF_PERCENT_N

union arg {
intintarg;
unsigned intuintarg;
longlongarg;
unsigned longulongarg;
long longlonglongarg;
unsigned long longulonglongarg;
ptrdiff_tptrdiffarg;
size_tsizearg;
ssize_tssizearg;
intmax_tintmaxarg;
uintmax_tuintmaxarg;
void*pvoidarg;
char*pchararg;
signed char*pschararg;
short*pshortarg;
int*pintarg;
long*plongarg;
long long*plonglongarg;
ptrdiff_t*pptrdiffarg;
ssize_t*pssizearg;
intmax_t*pintmaxarg;
#ifdef FLOATING_POINT
doubledoublearg;
long doublelongdoublearg;
#endif
#ifdef PRINTF_WIDE_CHAR
wint_twintarg;
wchar_t*pwchararg;
#endif
};

static int __find_arguments(const char *fmt0, va_list ap, union arg **argtable,
size_t *argtablesiz);

```

```

static int __grow_type_table(unsigned char **typetable, int *tablesize);
static int xxxprintf(char **, size_t, int, const char *, va_list);

#ifndef defined(MAP_ANON) && defined(MAP_ANONYMOUS)
#define MAP_ANON MAP_ANONYMOUS
#endif

/*
 * Allocate "size" bytes via mmap.
 */
static void *
mmap_alloc(size_t size)
{
    void *p;
#ifdef MAP_ANON
    int fd;

    if ((fd = open("/dev/zero", O_RDWR)) == -1)
        return NULL;
    p = mmap(NULL, size, PROT_READ | PROT_WRITE, MAP_PRIVATE, fd, 0);
    close(fd);
#else
    p = mmap(NULL, size, PROT_READ | PROT_WRITE, MAP_PRIVATE|MAP_ANON, -1, 0);
#endif
    if (p == MAP_FAILED)
        return NULL;
    return p;
}

/*
 * Unmap "size" bytes of the ptr.
 */
static void
mmap_free(void *ptr, size_t size)
{
    if (ptr != NULL)
        munmap(ptr, size);
}

#ifdef PRINTF_WIDE_CHAR
/*
 * Convert a wide character string argument for the %ls format to a multibyte
 * string representation. If not -1, prec specifies the maximum number of
 * bytes to output, and also means that we can't assume that the wide char
 * string is null-terminated.
 */
static char *
__wcsconv(wchar_t *wcsarg, int prec)
{
    mbstate_t mbs;
    char buf[MB_LEN_MAX];
    wchar_t *p;
    char *convbuf;
    size_t clen, nbytes;

```

```

/* Allocate space for the maximum number of bytes we could output. */
if (prec < 0) {
    memset(&mbs, 0, sizeof(mbs));
    p = wcsarg;
    nbytes = wcsrtombs(NULL, (const wchar_t **)&p, 0, &mbs);
    if (nbytes == (size_t)-1) {
        errno = EILSEQ;
        return NULL;
    }
} else {
    /*
     * Optimisation: if the output precision is small enough,
     * just allocate enough memory for the maximum instead of
     * scanning the string.
     */
    if (prec < 128)
        nbytes = prec;
    else {
        nbytes = 0;
        p = wcsarg;
        memset(&mbs, 0, sizeof(mbs));
        for (;;) {
            clen = wcrctomb(buf, *p++, &mbs);
            if (clen == 0 || clen == (size_t)-1 ||
                nbytes + clen > (size_t)prec)
                break;
            nbytes += clen;
        }
        if (clen == (size_t)-1) {
            errno = EILSEQ;
            return NULL;
        }
    }
}
if ((convbuf = malloc(nbytes + 1)) == NULL)
    return NULL;

/* Fill the output buffer. */
p = wcsarg;
memset(&mbs, 0, sizeof(mbs));
if ((nbytes = wcsrtombs(convbuf, (const wchar_t **)&p,
    nbytes, &mbs)) == (size_t)-1) {
    free(convbuf);
    errno = EILSEQ;
    return NULL;
}
convbuf[nbytes] = '\0';
return convbuf;
}
#endif

#ifdef FLOATING_POINT
#include <float.h>
#include <locale.h>

```

```

#include <math.h>
#include "floatio.h"
#include "gdtoa.h"

#define DEFPREC6

static int exponent(char *, int, int);
#endif /* FLOATING_POINT */

/*
 * The size of the buffer we use as scratch space for integer
 * conversions, among other things. Technically, we would need the
 * most space for base 10 conversions with thousands' grouping
 * characters between each pair of digits. 100 bytes is a
 * conservative overestimate even for a 128-bit uintmax_t.
 */
#define BUF100

#define STATIC_ARG_TBL_SIZE 8 /* Size of static argument table. */

/*
 * Macros for converting digits to letters and vice versa
 */
#define to_digit(c)((c) - '0')
#define is_digit(c)((unsigned int)to_digit(c) <= 9)
#define to_char(n)((n) + '0')

/*
 * Flags used during conversion.
 */
#define ALT0x0001 /* alternate form */
#define LADJUST0x0004 /* left adjustment */
#define LONGDBL0x0008 /* long double */
#define LONGINT0x0010 /* long integer */
#define LLONGINT0x0020 /* long long integer */
#define SHORTINT0x0040 /* short integer */
#define ZEROPAD0x0080 /* zero (as opposed to blank) pad */
#define FPT0x0100 /* Floating point number */
#define PTRINT0x0200 /* (unsigned) ptrdiff_t */
#define SIZEINT0x0400 /* (signed) size_t */
#define CHARINT0x0800 /* 8 bit integer */
#undef MAXINT /* Also defined by HP-UX param.h... */
#define MAXINT0x1000 /* largest integer size (intmax_t) */

/*
 * Actual printf innards.
 */
static int
xxxprintf(char **strp, size_t strsize, int alloc, const char *fmt0, va_list ap)
{
    char *fmt; /* format string */
    int ch; /* character from fmt */
    int n, n2; /* handy integers (short term usage) */
    char *cp; /* handy char pointer (short term usage) */

```

```

int flags; /* flags as above */
int ret; /* return value accumulator */
int width; /* width from format (%8d), or 0 */
int prec; /* precision from format; <0 for N/A */
char sign; /* sign prefix (' ', '+', '-', or \0) */
#ifdef FLOATING_POINT
/*
 * We can decompose the printed representation of floating
 * point numbers into several parts, some of which may be empty:
 *
 * [+|-] [0x|0X] MMM . NNN [e|E|p|P] [+|-] ZZ
 * A   B   ---C---   D   E F
 *
 * A: 'sign' holds this value if present; '\0' otherwise
 * B: ox[1] holds the 'x' or 'X'; '\0' if not hexadecimal
 * C: cp points to the string MMMNNN. Leading and trailing
 * zeros are not in the string and must be added.
 * D: expchar holds this character; '\0' if no exponent, e.g. %f
 * F: at least two digits for decimal, at least one digit for hex
 */
#ifdef HAVE_NL_LANGINFO
const char *decimal_point = NULL;
#else
const char *decimal_point = ".";
#endif
int signflag; /* true if float is negative */
union { /* floating point arguments %[aAeEfFgG] */
double dbl;
long double ldbl;
} fparg;
int expt; /* integer value of exponent */
char expchar; /* exponent character: [eEpP\0] */
char *dtoaend; /* pointer to end of converted digits */
int expsize; /* character count for expstr */
int lead; /* sig figs before decimal or group sep */
int ndig; /* actual number of digits returned by dtoa */
char expstr[MAXEXPDIG+2]; /* buffer for exponent string: e+ZZZ */
char *dtoaresult = NULL;
#endif

uintmax_t _umax; /* integer arguments %[diouxX] */
enum { OCT, DEC, HEX } base; /* base for %[diouxX] conversion */
int dprec; /* a copy of prec if %[diouxX], 0 otherwise */
int realsz; /* field size expanded by dprec */
int size; /* size of converted field or string */
const char *xdigs = ""; /* digits for %[xX] conversion */
#define NIOV 8
char buf[BUF]; /* buffer with space for digits of uintmax_t */
char ox[2]; /* space for 0x; ox[1] is either x, X, or \0 */
char *str; /* pointer to string to fill */
char *estr; /* pointer to last char in str */
union arg *argtable; /* args, built due to positional arg */
union arg statargtable[STATIC_ARG_TBL_SIZE];
size_t argtablesiz;

```

```

int nextarg; /* 1-based argument index */
va_list orgap; /* original argument pointer */
#ifdef PRINTF_WIDE_CHAR
char *convbuf; /* buffer for wide to multi-byte conversion */
#endif

/*
 * Choose PADSIZ to trade efficiency vs. size. If larger printf
 * fields occur frequently, increase PADSIZ and make the initialisers
 * below longer.
 */
#define PADSIZ 16 /* pad chunk size */
static char blanks[PADSIZ] =
    {' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '};
static char zeroes[PADSIZ] =
    {'0','0','0','0','0','0','0','0','0','0','0','0','0','0','0','0'};

static const char xdigs_lower[16] = "0123456789abcdef";
static const char xdigs_upper[16] = "0123456789ABCDEF";

/* Print chars to "str", (allocate as needed if alloc is set). */
#define PRINT(ptr, len) do { \
    const char *p = ptr; \
    const char *endp = ptr + len; \
    while (p < endp && (str < estr || alloc)) { \
        if (alloc && str >= estr) { \
            char *t; \
            strsize = (strsize << 1) + 1; \
            if (!(t = realloc(*strp, strsize))) { \
                free(str); \
                *strp = NULL; \
                ret = -1; \
                goto done; \
            } \
            str = t + (str - *strp); \
            estr = t + strsize - 1; \
            *strp = t; \
        } \
        *str++ = *p++; \
    } \
} while (0)

/* BEWARE, PAD uses `n' and PRINTANDPAD uses `n2'. */
#define PAD(plen, pstr) do { \
    if ((n = (plen)) > 0) { \
        while (n > PADSIZ) { \
            PRINT(pstr, PADSIZ); \
            n -= PADSIZ; \
        } \
        PRINT(pstr, n); \
    } \
} while (0)
#define PRINTANDPAD(p, ep, len, with) do { \
    n2 = (ep) - (p); \
    if (n2 > (len)) \

```



```

n2 = (len);\
if (n2 > 0)\
PRINT((p), n2);\
PAD((len) - (n2 > 0 ? n2 : 0), (with));\
} while(0)

/*
 * To extend shorts properly, we need both signed and unsigned
 * argument extraction methods.
 */
#define SARG() \
((intmax_t)(flags & MAXINT ? GETARG(intmax_t) : \
  flags & LLONGINT ? GETARG(long long) : \
  flags & LONGINT ? GETARG(long) : \
  flags & PTRINT ? GETARG(ptrdiff_t) : \
  flags & SIZEINT ? GETARG(ssize_t) : \
  flags & SHORTINT ? (short)GETARG(int) : \
  flags & CHARINT ? (signed char)GETARG(int) : \
  GETARG(int)))
#define UARG() \
((uintmax_t)(flags & MAXINT ? GETARG(uintmax_t) : \
  flags & LLONGINT ? GETARG(unsigned long long) : \
  flags & LONGINT ? GETARG(unsigned long) : \
  flags & PTRINT ? (uintptr_t)GETARG(ptrdiff_t) : /* XXX */ \
  flags & SIZEINT ? GETARG(size_t) : \
  flags & SHORTINT ? (unsigned short)GETARG(int) : \
  flags & CHARINT ? (unsigned char)GETARG(int) : \
  GETARG(unsigned int)))

/*
 * Append a digit to a value and check for overflow.
 */
#define APPEND_DIGIT(val, dig) do { \
if ((val) > INT_MAX / 10) \
goto overflow; \
(val) *= 10; \
if ((val) > INT_MAX - to_digit((dig))) \
goto overflow; \
(val) += to_digit((dig)); \
} while (0)

/*
 * Get * arguments, including the form *nn$. Preserve the nextarg
 * that the argument can be gotten once the type is determined.
 */
#define GETASTER(val) \
n2 = 0; \
cp = fmt; \
while (is_digit(*cp)) { \
APPEND_DIGIT(n2, *cp); \
cp++; \
} \
if (*cp == '$') { \
int hold = nextarg; \

```

```

if (argtable == NULL) { \
argtable = statargtable; \
__find_arguments(fmt0, orgap, &argtable, &argtablesiz); \
} \
nextarg = n2; \
val = GETARG(int); \
nextarg = hold; \
fmt = ++cp; \
} else { \
val = GETARG(int); \
}
}

/*
 * Get the argument indexed by nextarg.  If the argument table is
 * built, use it to get the argument.  If its not, get the next
 * argument (and arguments must be gotten sequentially).
 */
#define GETARG(type) \
((argtable != NULL) ? *((type*)&argtable[nextarg++]) : \
(nextarg++, va_arg(ap, type)))

fmt = (char *)fmt0;
argtable = NULL;
nextarg = 1;
va_copy(orgap, ap);
ret = 0;
#ifdef PRINTF_WIDE_CHAR
convbuf = NULL;
#endif

if (alloc) {
strsize = 128;
*strp = str = malloc(strsize);
if (str == NULL) {
ret = -1;
goto done;
}
estr = str + 127;
} else {
str = *strp;
if (strsize)
estr = str + strsize - 1;
else
estr = NULL;
}

/*
 * Scan the format for conversions ('%' character).
 */
for (;;) {
for (cp = fmt; (ch = *fmt) != '\0' && ch != '%'; fmt++)
/* void */;
if ((n = fmt - cp) != 0) {
if (n > INT_MAX - ret)
goto overflow;

```

```
PRINT(cp, n);
ret += n;
}
if (ch == '\0')
goto done;
fmt++;/* skip over '%' */

flags = 0;
dprec = 0;
width = 0;
prec = -1;
sign = '\0';
ox[1] = '\0';

rflag:ch = *fmt++;
reswitch:switch (ch) {
case ' ':
/*
 * ``If the space and + flags both appear, the space
 * flag will be ignored."
 *-- ANSI X3J11
 */
if (!sign)
sign = ' ';
goto rflag;
case '#':
flags |= ALT;
goto rflag;
case '\':
/* grouping not implemented */
goto rflag;
case '*':
/*
 * ``A negative field width argument is taken as a
 * - flag followed by a positive field width."
 *-- ANSI X3J11
 * They don't exclude field widths read from args.
 */
GETASTER(width);
if (width >= 0)
goto rflag;
if (width == INT_MIN)
goto overflow;
width = -width;
/* FALLTHROUGH */
case '-':
flags |= LADJUST;
goto rflag;
case '+':
sign = '+';
goto rflag;
case '!':
if ((ch = *fmt++) == '*') {
GETASTER(n);
```

```

prec = n < 0 ? -1 : n;
goto rflag;
}
n = 0;
while (is_digit(ch)) {
APPEND_DIGIT(n, ch);
ch = *fmt++;
}
if (ch == '$') {
nextarg = n;
if (argtable == NULL) {
argtable = statargtable;
__find_arguments(fmt0, orgap,
&argtable, &argtablesiz);
}
goto rflag;
}
prec = n;
goto reswitch;
case '0':
/*
 * ``Note that 0 is taken as a flag, not as the
 * beginning of a field width."
 *-- ANSI X3J11
 */
flags |= ZEROPAD;
goto rflag;
case '1': case '2': case '3': case '4':
case '5': case '6': case '7': case '8': case '9':
n = 0;
do {
APPEND_DIGIT(n, ch);
ch = *fmt++;
} while (is_digit(ch));
if (ch == '$') {
nextarg = n;
if (argtable == NULL) {
argtable = statargtable;
__find_arguments(fmt0, orgap,
&argtable, &argtablesiz);
}
goto rflag;
}
width = n;
goto reswitch;
#ifdef FLOATING_POINT
case 'L':
flags |= LONGDBL;
goto rflag;
#endif
case 'h':
if (*fmt == 'h') {
fmt++;
flags |= CHARINT;
} else {

```

```
    flags |= SHORTINT;
}
goto rflag;
case 'j':
    flags |= MAXINT;
    goto rflag;
case 'l':
    if (*fmt == 'l') {
        fmt++;
        flags |= LLONGINT;
    } else {
        flags |= LONGINT;
    }
    goto rflag;
case 'q':
    flags |= LLONGINT;
    goto rflag;
case 't':
    flags |= PTRINT;
    goto rflag;
case 'z':
    flags |= SIZEINT;
    goto rflag;
case 'c':
#ifdef PRINTF_WIDE_CHAR
    if (flags & LONGINT) {
        mbstate_t mbs;
        size_t mbseqlen;

        memset(&mbs, 0, sizeof(mbs));
        mbseqlen = wcrctomb(buf,
            (wchar_t)GETARG(wint_t), &mbs);
        if (mbseqlen == (size_t)-1) {
            errno = EILSEQ;
            goto done;
        }
        cp = buf;
        size = (int)mbseqlen;
    } else {
#endif
        *(cp = buf) = GETARG(int);
        size = 1;
#ifdef PRINTF_WIDE_CHAR
    }
#endif
    sign = '\0';
    break;
case 'D':
    flags |= LONGINT;
    /*FALLTHROUGH*/
case 'd':
case 'i':
    _umax = SARG();
    if ((intmax_t)_umax < 0) {
```

```

    _umax = -_umax;
    sign = '-';
}
base = DEC;
goto number;
#ifdef FLOATING_POINT
case 'a':
case 'A':
if (ch == 'a') {
    ox[1] = 'x';
    xdigs = xdigs_lower;
    expchar = 'p';
} else {
    ox[1] = 'X';
    xdigs = xdigs_upper;
    expchar = 'P';
}
if (prec >= 0)
    prec++;
if (dtoaresult)
    __freedtoa(dtoaresult);
if (flags & LONGDBL) {
    fparg.ldbl = GETARG(long double);
    dtoaresult = cp =
        __hldtoa(fparg.ldbl, xdigs, prec,
            &expt, &signflag, &dtoaend);
if (dtoaresult == NULL) {
    errno = ENOMEM;
    goto done;
}
} else {
    fparg.dbl = GETARG(double);
    dtoaresult = cp =
        __hldtoa(fparg.dbl, xdigs, prec,
            &expt, &signflag, &dtoaend);
if (dtoaresult == NULL) {
    errno = ENOMEM;
    goto done;
}
}
if (prec < 0)
    prec = dtoaend - cp;
if (expt == INT_MAX)
    ox[1] = '\0';
goto fp_common;
case 'e':
case 'E':
    expchar = ch;
if (prec < 0)/* account for digit before decpt */
    prec = DEFPREC + 1;
else
    prec++;
goto fp_begin;
case 'f':
case 'F':

```

```

expchar = '\0';
goto fp_begin;
case 'g':
case 'G':
expchar = ch - ('g' - 'e');
if (prec == 0)
    prec = 1;
fp_begin:
if (prec < 0)
    prec = DEFPREC;
if (dtoaresult)
    __freedtoa(dtoaresult);
if (flags & LONGDBL) {
    fparg.ldbl = GETARG(long double);
    dtoaresult = cp =
        __ldtoa(&fparg.ldbl, expchar ? 2 : 3, prec,
            &expt, &signflag, &dtoaend);
if (dtoaresult == NULL) {
    errno = ENOMEM;
    goto done;
}
} else {
    fparg.dbl = GETARG(double);
    dtoaresult = cp =
        __dtoa(fparg.dbl, expchar ? 2 : 3, prec,
            &expt, &signflag, &dtoaend);
if (dtoaresult == NULL) {
    errno = ENOMEM;
    goto done;
}
if (expt == 9999)
    expt = INT_MAX;
}
fp_common:
if (signflag)
    sign = '-';
if (expt == INT_MAX) /* inf or nan */
if (*cp == 'N')
    cp = (ch >= 'a') ? "nan" : "NAN";
else
    cp = (ch >= 'a') ? "inf" : "INF";
    size = 3;
    flags &= ~ZEROPAD;
    break;
}
flags |= FPT;
ndig = dtoaend - cp;
if (ch == 'g' || ch == 'G') {
if (expt > -4 && expt <= prec) {
/* Make %[gG] smell like %[fF] */
expchar = '\0';
if (flags & ALT)
    prec -= expt;
else

```

```

prec = ndig - expt;
if (prec < 0)
prec = 0;
} else {
/*
 * Make %[gG] smell like %[eE], but
 * trim trailing zeroes if no # flag.
 */
if (!(flags & ALT))
prec = ndig;
}
}
if (expchar) {
expsize = exponent(expstr, expt - 1, expchar);
size = expsize + prec;
if (prec > 1 || flags & ALT)
++size;
} else {
/* space for digits before decimal point */
if (expt > 0)
size = expt;
else/* "0" */
size = 1;
/* space for decimal pt and following digits */
if (prec || flags & ALT)
size += prec + 1;
lead = expt;
}
break;
#endif /* FLOATING_POINT */
#ifndef NO_PRINTF_PERCENT_N
case 'n':
if (flags & LLONGINT)
*GETARG(long long *) = ret;
else if (flags & LONGINT)
*GETARG(long *) = ret;
else if (flags & SHORTINT)
*GETARG(short *) = ret;
else if (flags & CHARINT)
*GETARG(signed char *) = ret;
else if (flags & PTRINT)
*GETARG(ptrdiff_t *) = ret;
else if (flags & SIZEINT)
*GETARG(ssize_t *) = ret;
else if (flags & MAXINT)
*GETARG(intmax_t *) = ret;
else
*GETARG(int *) = ret;
continue;/* no output */
#endif /* NO_PRINTF_PERCENT_N */
case 'O':
flags |= LONGINT;
/*FALLTHROUGH*/
case 'o':
_uamax = UARG();

```



```

base = OCT;
goto nosign;
case 'p':
/*
 * ``The argument shall be a pointer to void. The
 * value of the pointer is converted to a sequence
 * of printable characters, in an implementation-
 * defined manner."
 *-- ANSI X3J11
 */
/* NOSTRICT */
_u_max = (u_long)GETARG(void *);
base = HEX;
xdigs = xdigs_lower;
ox[1] = 'x';
goto nosign;
case 's':
#ifdef PRINTF_WIDE_CHAR
if (flags & LONGINT) {
wchar_t *wcp;

if (convbuf != NULL) {
free(convbuf);
convbuf = NULL;
}
if ((wcp = GETARG(wchar_t *)) == NULL) {
cp = "(null)";
} else {
convbuf = __wcsconv(wcp, prec);
if (convbuf == NULL)
goto done;
cp = convbuf;
}
} else
#endif /* PRINTF_WIDE_CHAR */
if ((cp = GETARG(char *)) == NULL)
cp = "(null)";
if (prec >= 0) {
/*
 * can't use strlen; can only look for the
 * NUL in the first `prec' characters, and
 * strlen() will go further.
 */
char *p = memchr(cp, 0, prec);

size = p ? (p - cp) : prec;
} else {
size_t len;

if ((len = strlen(cp)) > INT_MAX)
goto overflow;
size = (int)len;
}
sign = '\0';

```

```

break;
case 'U':
    flags |= LONGINT;
    /*FALLTHROUGH*/
case 'u':
    _umax = UARG();
    base = DEC;
    goto nosign;
case 'X':
    xdigs = xdigs_upper;
    goto hex;
case 'x':
    xdigs = xdigs_lower;
hex: _umax = UARG();
    base = HEX;
    /* leading 0x/X only if non-zero */
    if (flags & ALT && _umax != 0)
        ox[1] = ch;

    /* unsigned conversions */
    nosign: sign = '\0';
    /*
     * ``... diouXx conversions ... if a precision is
     * specified, the 0 flag will be ignored."
     *-- ANSI X3J11
     */
    number: if ((dprec = prec) >= 0)
        flags &= ~ZEROPAD;

    /*
     * ``The result of converting a zero value with an
     * explicit precision of zero is no characters."
     *-- ANSI X3J11
     */
    cp = buf + BUF;
    if (_umax != 0 || prec != 0) {
    /*
     * Unsigned mod is hard, and unsigned mod
     * by a constant is easier than that by
     * a variable; hence this switch.
     */
        switch (base) {
        case OCT:
            do {
                *--cp = to_char(_umax & 7);
                _umax >>= 3;
            } while (_umax);
            /* handle octal leading 0 */
            if (flags & ALT && *cp != '0')
                *--cp = '0';
            break;

        case DEC:
            /* many numbers are 1 digit */
            while (_umax >= 10) {

```

```

*--cp = to_char(_umax % 10);
_umax /= 10;
}
*--cp = to_char(_umax);
break;

case HEX:
do {
*--cp = xdigs[_umax & 15];
_umax >>= 4;
} while (_umax);
break;

default:
cp = "bug in fprintf: bad base";
size = strlen(cp);
goto skipsize;
}
}
size = buf + BUF - cp;
if (size > BUF)/* should never happen */
abort();
skipsize:
break;
default:/* "%?" prints ?, unless ? is NUL */
if (ch == '\0')
goto done;
/* pretend it was %c with argument ch */
cp = buf;
*cp = ch;
size = 1;
sign = '\0';
break;
}

/*
 * All reasonable formats wind up here. At this point, `cp'
 * points to a string which (if not flags&LADJUST) should be
 * padded out to `width' places. If flags&ZEROPAD, it should
 * first be prefixed by any sign or other prefix; otherwise,
 * it should be blank padded before the prefix is emitted.
 * After any left-hand padding and prefixing, emit zeroes
 * required by a decimal %[diouxX] precision, then print the
 * string proper, then emit zeroes required by any leftover
 * floating precision; finally, if LADJUST, pad with blanks.
 *
 * Compute actual size, so we know how much to pad.
 * size excludes decimal prec; realsz includes it.
 */
realsz = dprec > size ? dprec : size;
if (sign)
realsz++;
if (ox[1])
realsz+= 2;

```

```

/* right-adjusting blank padding */
if ((flags & (LADJUST|ZEROPAD)) == 0)
    PAD(width - realsz, blanks);

/* prefix */
if (sign)
    PRINT(&sign, 1);
if (ox[1]) { /* ox[1] is either x, X, or \0 */
    ox[0] = '0';
    PRINT(ox, 2);
}

/* right-adjusting zero padding */
if ((flags & (LADJUST|ZEROPAD)) == ZEROPAD)
    PAD(width - realsz, zeroes);

/* leading zeroes from decimal precision */
PAD(dprec - size, zeroes);

/* the string or number proper */
#ifdef FLOATING_POINT
if ((flags & FPT) == 0) {
    PRINT(cp, size);
} else { /* glue together f_p fragments */
#ifdef HAVE_NL_LANGINFO
if (decimal_point == NULL)
    decimal_point = nl_langinfo(RADIXCHAR);
#endif
if (!expchar) { /* %[fF] or sufficiently short %[gG] */
if (expt <= 0) {
    PRINT(zeroes, 1);
if (prec || flags & ALT)
    PRINT(decimal_point, 1);
    PAD(-expt, zeroes);
/* already handled initial 0's */
    prec += expt;
} else {
    PRINTANDPAD(cp, dtoaend, lead, zeroes);
    cp += lead;
if (prec || flags & ALT)
    PRINT(decimal_point, 1);
}
    PRINTANDPAD(cp, dtoaend, prec, zeroes);
} else { /* %[eE] or sufficiently long %[gG] */
if (prec > 1 || flags & ALT) {
    buf[0] = *cp++;
    buf[1] = *decimal_point;
    PRINT(buf, 2);
    PRINT(cp, ndig-1);
    PAD(prec - ndig, zeroes);
} else { /* XeYYY */
    PRINT(cp, 1);
}
    PRINT(expstr, expsize);
}

```

```
}
}
#else
PRINT(cp, size);
#endif
/* left-adjusting padding (always blank) */
if (flags & LADJUST)
PAD(width - realsz, blanks);

/* finally, adjust ret */
if (width < realsz)
width = realsz;
if (width > INT_MAX - ret)
goto overflow;
ret += width;
}
done:
va_end(orgap);
if (strsize)
*str = '\0';
goto finish;

overflow:
errno = EOVERFLOW;
ret = -1;

finish:
#ifdef PRINTF_WIDE_CHAR
if (convbuf)
free(convbuf);
#endif
#ifdef FLOATING_POINT
if (dtoaresult)
__freedtoa(dtoaresult);
#endif
if (argtable != NULL && argtable != statargtable) {
mmap_free(argtable, argtablesiz);
argtable = NULL;
}
return ret;
}

/*
 * Type ids for argument type table.
 */
#define T_UNUSED0
#define T_SHORT1
#define T_U_SHORT2
#define TP_SHORT3
#define T_INT4
#define T_U_INT5
#define TP_INT6
#define T_LONG7
#define T_U_LONG8
```

```

#define TP_LONG9
#define T_LLONG10
#define T_U_LLONG11
#define TP_LLONG12
#define T_DOUBLE13
#define T_LONG_DOUBLE14
#define TP_CHAR15
#define TP_VOID16
#define T_PTRINT17
#define TP_PTRINT18
#define T_SIZEINT19
#define T_SSIZEINT20
#define TP_SSIZEINT21
#define T_MAXINT22
#define T_MAXUINT23
#define TP_MAXINT24
#define T_CHAR25
#define T_U_CHAR26
#define T_WINT27
#define TP_WCHAR28

/*
 * Find all arguments when a positional parameter is encountered. Returns a
 * table, indexed by argument number, of pointers to each arguments. The
 * initial argument table should be an array of STATIC_ARG_TBL_SIZE entries.
 * It will be replaced with a mmap-ed one if it overflows (malloc cannot be
 * used since we are attempting to make snprintf thread safe, and alloca is
 * problematic since we have nested functions..)
 */
static int
__find_arguments(const char *fmt0, va_list ap, union arg **argtable,
                 size_t *argtablesiz)
{
    char *fmt; /* format string */
    int ch; /* character from fmt */
    int n, n2; /* handy integer (short term usage) */
    char *cp; /* handy char pointer (short term usage) */
    int flags; /* flags as above */
    unsigned char *typetable; /* table of types */
    unsigned char stattypetable[STATIC_ARG_TBL_SIZE];
    int tablesiz; /* current size of type table */
    int tablemax; /* largest used index in table */
    int nextarg; /* 1-based argument index */
    int ret = 0; /* return value */

    /*
     * Add an argument type to the table, expanding if necessary.
     */
    #define ADDTYPE(type) \
        ((nextarg >= tablesiz) ? \
         __grow_type_table(&typetable, &tablesiz) : 0, \
        (nextarg > tablemax) ? tablemax = nextarg : 0, \
        typetable[nextarg++] = type)

    #define ADDSARG() \

```

```

        ((flags&MAXINT) ? ADDTYPE(T_MAXINT) : \
        ((flags&PTRINT) ? ADDTYPE(T_PTRINT) : \
        ((flags&SIZEINT) ? ADDTYPE(T_SSIZEINT) : \
        ((flags&LLONGINT) ? ADDTYPE(T_LLONG) : \
        ((flags&LONGINT) ? ADDTYPE(T_LONG) : \
        ((flags&SHORTINT) ? ADDTYPE(T_SHORT) : \
        ((flags&CHARINT) ? ADDTYPE(T_CHAR) : ADDTYPE(T_INT))))))

#defineADDUARG() \
    ((flags&MAXINT) ? ADDTYPE(T_MAXUINT) : \
    ((flags&PTRINT) ? ADDTYPE(T_PTRINT) : \
    ((flags&SIZEINT) ? ADDTYPE(T_SIZEINT) : \
    ((flags&LLONGINT) ? ADDTYPE(T_U_LLONG) : \
    ((flags&LONGINT) ? ADDTYPE(T_U_LONG) : \
    ((flags&SHORTINT) ? ADDTYPE(T_U_SHORT) : \
    ((flags&CHARINT) ? ADDTYPE(T_U_CHAR) : ADDTYPE(T_U_INT))))))

/*
 * Add * arguments to the type array.
 */
#define ADDASTER() \
n2 = 0; \
cp = fmt; \
while (is_digit(*cp)) { \
APPEND_DIGIT(n2, *cp); \
cp++; \
} \
if (*cp == '$') { \
int hold = nextarg; \
nextarg = n2; \
ADDTYPE(T_INT); \
nextarg = hold; \
fmt = ++cp; \
} else { \
ADDTYPE(T_INT); \
} \
fmt = (char *)fmt0; \
typetable = stattypetable; \
tablesize = STATIC_ARG_TBL_SIZE; \
tablemax = 0; \
nextarg = 1; \
memset(typetable, T_UNUSED, STATIC_ARG_TBL_SIZE);

/*
 * Scan the format for conversions ('%' character).
 */
for (;;) {
for (cp = fmt; (ch = *fmt) != '\0' && ch != '%'; fmt++)
/* void */;
fmt++;/* skip over '%' */

flags = 0;

rflag:ch = *fmt++;

```

```

reswitch:switch (ch) {
case ' ':
case '#':
case '\':
goto rflag;
case '*':
ADDASTER();
goto rflag;
case '-':
case '+':
goto rflag;
case '.':
if ((ch = *fmt++) == '*') {
ADDASTER();
goto rflag;
}
while (is_digit(ch)) {
ch = *fmt++;
}
goto reswitch;
case '0':
goto rflag;
case '1': case '2': case '3': case '4':
case '5': case '6': case '7': case '8': case '9':
n = 0;
do {
APPEND_DIGIT(n ,ch);
ch = *fmt++;
} while (is_digit(ch));
if (ch == '$') {
nextarg = n;
goto rflag;
}
goto reswitch;
#ifdef FLOATING_POINT
case 'L':
flags |= LONGDBL;
goto rflag;
#endif
case 'h':
if (*fmt == 'h') {
fmt++;
flags |= CHARINT;
} else {
flags |= SHORTINT;
}
goto rflag;
case 'j':
flags |= MAXINT;
goto rflag;
case 'l':
if (*fmt == 'l') {
fmt++;
flags |= LLONGINT;
} else {

```



```
    flags |= LONGINT;
}
goto rflag;
case 'q':
    flags |= LLONGINT;
    goto rflag;
case 't':
    flags |= PTRINT;
    goto rflag;
case 'z':
    flags |= SIZEINT;
    goto rflag;
case 'c':
#ifdef PRINTF_WIDE_CHAR
    if (flags & LONGINT)
        ADDTYPE(T_WINT);
    else
#endif
        ADDTYPE(T_INT);
    break;
case 'D':
    flags |= LONGINT;
    /*FALLTHROUGH*/
case 'd':
case 'i':
    ADDSARG();
    break;
#ifdef FLOATING_POINT
case 'a':
case 'A':
case 'e':
case 'E':
case 'f':
case 'F':
case 'g':
case 'G':
    if (flags & LONGDBL)
        ADDTYPE(T_LONG_DOUBLE);
    else
        ADDTYPE(T_DOUBLE);
    break;
#endif /* FLOATING_POINT */
#ifndef NO_PRINTF_PERCENT_N
case 'n':
    if (flags & LLONGINT)
        ADDTYPE(TP_LLONG);
    else if (flags & LONGINT)
        ADDTYPE(TP_LONG);
    else if (flags & SHORTINT)
        ADDTYPE(TP_SHORT);
    else if (flags & PTRINT)
        ADDTYPE(TP_PTRINT);
    else if (flags & SIZEINT)
        ADDTYPE(TP_SSIZEINT);
```

```

else if (flags & MAXINT)
    ADDTYPE(TP_MAXINT);
else
    ADDTYPE(TP_INT);
continue; /* no output */
#endif /* NO_PRINTF_PERCENT_N */
case 'O':
    flags |= LONGINT;
    /*FALLTHROUGH*/
case 'o':
    ADDUARG();
    break;
case 'p':
    ADDTYPE(TP_VOID);
    break;
case 's':
#ifdef PRINTF_WIDE_CHAR
    if (flags & LONGINT)
        ADDTYPE(TP_WCHAR);
    else
#endif
        ADDTYPE(TP_CHAR);
    break;
case 'U':
    flags |= LONGINT;
    /*FALLTHROUGH*/
case 'u':
case 'X':
case 'x':
    ADDUARG();
    break;
default: /* "%?" prints ?, unless ? is NUL */
    if (ch == '\0')
        goto done;
    break;
}
}
done:
/*
 * Build the argument table.
 */
if (tablemax >= STATIC_ARG_TBL_SIZE) {
    *argtablesiz = sizeof(union arg) * (tablemax + 1);
    *argtable = mmap_alloc(*argtablesiz);
    if (*argtable == NULL)
        return -1;
}

for (n = 1; n <= tablemax; n++) {
    switch (typetable[n]) {
    case T_UNUSED:
    case T_CHAR:
    case T_U_CHAR:
    case T_SHORT:
    case T_U_SHORT:

```

```
case T_INT:
(*argtable)[n].intarg = va_arg(ap, int);
break;
case TP_SHORT:
(*argtable)[n].pshortarg = va_arg(ap, short *);
break;
case T_U_INT:
(*argtable)[n].uintarg = va_arg(ap, unsigned int);
break;
case TP_INT:
(*argtable)[n].pintarg = va_arg(ap, int *);
break;
case T_LONG:
(*argtable)[n].longarg = va_arg(ap, long);
break;
case T_U_LONG:
(*argtable)[n].ulongarg = va_arg(ap, unsigned long);
break;
case TP_LONG:
(*argtable)[n].plongarg = va_arg(ap, long *);
break;
case T_LLONG:
(*argtable)[n].longlongarg = va_arg(ap, long long);
break;
case T_U_LLONG:
(*argtable)[n].ulonglongarg = va_arg(ap, unsigned long long);
break;
case TP_LLONG:
(*argtable)[n].plonglongarg = va_arg(ap, long long *);
break;
#ifdef FLOATING_POINT
case T_DOUBLE:
(*argtable)[n].doublearg = va_arg(ap, double);
break;
case T_LONG_DOUBLE:
(*argtable)[n].longdoublearg = va_arg(ap, long double);
break;
#endif
case TP_CHAR:
(*argtable)[n].pchararg = va_arg(ap, char *);
break;
case TP_VOID:
(*argtable)[n].pvoidarg = va_arg(ap, void *);
break;
case T_PTRINT:
(*argtable)[n].ptrdiffarg = va_arg(ap, ptrdiff_t);
break;
case TP_PTRINT:
(*argtable)[n].pptrdiffarg = va_arg(ap, ptrdiff_t *);
break;
case T_SIZEINT:
(*argtable)[n].sizearg = va_arg(ap, size_t);
break;
case T_SSIZEINT:
```

```

(*argtable)[n].ssizearg = va_arg(ap, ssize_t);
break;
case TP_SSIZEINT:
(*argtable)[n].pssizearg = va_arg(ap, ssize_t *);
break;
case T_MAXINT:
(*argtable)[n].intmaxarg = va_arg(ap, intmax_t);
break;
case T_MAXUINT:
(*argtable)[n].uintmaxarg = va_arg(ap, uintmax_t);
break;
case TP_MAXINT:
(*argtable)[n].pintmaxarg = va_arg(ap, intmax_t *);
break;
#ifdef PRINTF_WIDE_CHAR
case T_WINT:
(*argtable)[n].wintarg = va_arg(ap, wint_t);
break;
case TP_WCHAR:
(*argtable)[n].pwchararg = va_arg(ap, wchar_t *);
break;
#endif
}
}
goto finish;

overflow:
errno = EOVERFLOW;
ret = -1;

finish:
if (typetable != NULL && typetable != stattypetable) {
mmap_free(typetable, *argtablesiz);
typetable = NULL;
}
return ret;
}

/*
 * Increase the size of the type table.
 */
static int
__grow_type_table(unsigned char **typetable, int *tablesize)
{
unsigned char *oldtable = *typetable;
int newsize = *tablesize * 2;

if (newsize < sysconf(_SC_PAGESIZE))
newsize = sysconf(_SC_PAGESIZE);

if (*tablesize == STATIC_ARG_TBL_SIZE) {
*typetable = mmap_alloc(newsize);
if (*typetable == NULL)
return -1;
memcpy(*typetable, oldtable, *tablesize);

```

```

    } else {
        unsigned char *new = mmap_alloc(newsize);
        if (new == NULL)
            return -1;
        memmove(new, *typetable, *tablesize);
        mmap_free(*typetable, *tablesize);
        *typetable = new;
    }
    memset(*typetable + *tablesize, T_UNUSED, (newsize - *tablesize));

    *tablesize = newsize;
    return 0;
}

#ifdef FLOATING_POINT
static int
exponent(char *p0, int exp, int fmtch)
{
    char *p, *t;
    char expbuf[MAXEXPDIG];

    p = p0;
    *p++ = fmtch;
    if (exp < 0) {
        exp = -exp;
        *p++ = '-';
    } else
        *p++ = '+';
    t = expbuf + MAXEXPDIG;
    if (exp > 9) {
        do {
            *--t = to_char(exp % 10);
        } while ((exp /= 10) > 9);
        *--t = to_char(exp);
        for (; t < expbuf + MAXEXPDIG; *p++ = *t++)
            /* nothing */;
    } else {
        /*
         * Exponents for decimal floating point conversions
         * (%[eEgG]) must be at least two characters long,
         * whereas exponents for hexadecimal conversions can
         * be only one character long.
         */
        if (fmtch == 'e' || fmtch == 'E')
            *p++ = '0';
        *p++ = to_char(exp);
    }
    return p - p0;
}
#endif /* FLOATING_POINT */

#if !defined(HAVE_VSNPRINTF) || defined(PREFER_PORTABLE_SNPRINTF)
int

```

```

sudo_vsnprintf(char *str, size_t n, const char *fmt, va_list ap)
{
    if (n > INT_MAX) {
        errno = EOVERFLOW;
        *str = '\0';
        return -1;
    }
    return xxxprintf(&str, n, 0, fmt, ap);
}
#endif /* !HAVE_VSNPRINTF || PREFER_PORTABLE_SNPRINTF */

#if !defined(HAVE_SNPRINTF) || defined(PREFER_PORTABLE_SNPRINTF)
int
sudo_snprintf(char *str, size_t n, char const *fmt, ...)
{
    int ret;
    va_list ap;

    if (n > INT_MAX) {
        errno = EOVERFLOW;
        *str = '\0';
        return -1;
    }
    va_start(ap, fmt);
    ret = xxxprintf(&str, n, 0, fmt, ap);
    va_end(ap);
    return ret;
}
#endif /* !HAVE_SNPRINTF || PREFER_PORTABLE_SNPRINTF */

#if !defined(HAVE_VASPRINTF) || defined(PREFER_PORTABLE_SNPRINTF)
int
sudo_vasprintf(char **str, const char *fmt, va_list ap)
{
    int ret;

    ret = xxxprintf(str, 0, 1, fmt, ap);
    if (ret == -1)
        *str = NULL;
    return ret;
}
#endif /* !HAVE_VASPRINTF || PREFER_PORTABLE_SNPRINTF */

#if !defined(HAVE_ASPRINTF) || defined(PREFER_PORTABLE_SNPRINTF)
int
sudo_asprintf(char **str, char const *fmt, ...)
{
    int ret;
    va_list ap;

    va_start(ap, fmt);
    ret = xxxprintf(str, 0, 1, fmt, ap);
    va_end(ap);
    if (ret == -1)
        *str = NULL;
}

```

```

return ret;
}
#endif /* !HAVE_ASPRINTF || PREFER_PORTABLE_SNPRINTF */

#endif /* !HAVE_VSNPRINTF || !HAVE_SNPRINTF || !HAVE_VASPRINTF || !HAVE_ASPRINTF ||
PREFER_PORTABLE_SNPRINTF */
/*
 * Copyright (c) 1991, 1993
 * The Regents of the University of California. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the University nor the names of its contributors
 * may be used to endorse or promote products derived from this software
 * without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 * @(#)queue.h8.5 (Berkeley) 8/20/94
 * $FreeBSD: head/sys/sys/queue.h 251887 2013-06-18 02:57:56Z lstewart $
 */

#ifndef SUDO_QUEUE_H
#define SUDO_QUEUE_H

/*
 * This file defines four types of data structures: singly-linked lists,
 * singly-linked tail queues, lists and tail queues.
 *
 * A singly-linked list is headed by a single forward pointer. The elements
 * are singly linked for minimum space and pointer manipulation overhead at
 * the expense of O(n) removal for arbitrary elements. New elements can be
 * added to the list after an existing element or at the head of the list.
 * Elements being removed from the head of the list should use the explicit
 * macro for this purpose for optimum efficiency. A singly-linked list may
 * only be traversed in the forward direction. Singly-linked lists are ideal
 * for applications with large datasets and few or no removals or for
 * implementing a LIFO queue.

```

*
 * A singly-linked tail queue is headed by a pair of pointers, one to the
 * head of the list and the other to the tail of the list. The elements are
 * singly linked for minimum space and pointer manipulation overhead at the
 * expense of $O(n)$ removal for arbitrary elements. New elements can be added
 * to the list after an existing element, at the head of the list, or at the
 * end of the list. Elements being removed from the head of the tail queue
 * should use the explicit macro for this purpose for optimum efficiency.
 * A singly-linked tail queue may only be traversed in the forward direction.
 * Singly-linked tail queues are ideal for applications with large datasets
 * and few or no removals or for implementing a FIFO queue.
 *
 * A list is headed by a single forward pointer (or an array of forward
 * pointers for a hash table header). The elements are doubly linked
 * so that an arbitrary element can be removed without a need to
 * traverse the list. New elements can be added to the list before
 * or after an existing element or at the head of the list. A list
 * may be traversed in either direction.
 *
 * A tail queue is headed by a pair of pointers, one to the head of the
 * list and the other to the tail of the list. The elements are doubly
 * linked so that an arbitrary element can be removed without a need to
 * traverse the list. New elements can be added to the list before or
 * after an existing element, at the head of the list, or at the end of
 * the list. A tail queue may be traversed in either direction.
 *
 * A headless tail queue lacks a head structure, The first element acts
 * as a de facto list head. It uses the same entry struct as a regular
 * tail queue for easy conversion from headless to headful.
 * It is capable of concatenating queues as well as individual elements.
 * Traversing in reverse is more expensive due to lack of a list head.
 * Note: elements must be initialized before use.
 *
 * For details on the use of these macros, see the queue(3) manual page.
 *
 *
 *SLISTLISTSTAILQTAILQ
 * _HEAD++++
 * _HEAD_INITIALIZER++++
 * _ENTRY++++
 * _INIT++++
 * _EMPTY++++
 * _FIRST++++
 * _NEXT++++
 * _PREV-+-+
 * _LAST--++
 * _FOREACH++++
 * _FOREACH_FROM++++
 * _FOREACH_SAFE++++
 * _FOREACH_FROM_SAFE++++
 * _FOREACH_REVERSE---+
 * _FOREACH_REVERSE_FROM---+
 * _FOREACH_REVERSE_SAFE---+
 * _FOREACH_REVERSE_FROM_SAFE---+
 * _INSERT_HEAD++++


```

* _INSERT_BEFORE-+-+
* _INSERT_AFTER++++
* _INSERT_TAIL--++
* _CONCAT--++
* _REMOVE_AFTER+-+
* _REMOVE_HEAD+-+
* _REMOVE++++
* _SWAP++++
*
*/
#ifdef QUEUE_MACRO_DEBUG
/* Store the last 2 places the queue element or head was altered */
struct qm_trace {
    unsigned long lastline;
    unsigned long prevline;
    const char*lastfile;
    const char*prevfile;
};

#undef TRACEBUF
#defineTRACEBUFstruct qm_trace trace;
#undef TRACEBUF_INITIALIZER
#defineTRACEBUF_INITIALIZER{ __FILE__, __LINE__, NULL, 0 } ,
#undef TRASHIT
#defineTRASHIT(x)do {(x) = (void *)-1;} while (0)
#undef QMD_SAVELINK
#defineQMD_SAVELINK(name, link)void **name = (void *)&(link)

#undef QMD_TRACE_HEAD
#defineQMD_TRACE_HEAD(head) do {\
    (head)->trace.prevline = (head)->trace.lastline;\
    (head)->trace.prevfile = (head)->trace.lastfile;\
    (head)->trace.lastline = __LINE__;\
    (head)->trace.lastfile = __FILE__;\
} while (0)

#undef QMD_TRACE_ELEM
#defineQMD_TRACE_ELEM(elem) do {\
    (elem)->trace.prevline = (elem)->trace.lastline;\
    (elem)->trace.prevfile = (elem)->trace.lastfile;\
    (elem)->trace.lastline = __LINE__;\
    (elem)->trace.lastfile = __FILE__;\
} while (0)

#else
#undef QMD_TRACE_ELEM
#defineQMD_TRACE_ELEM(elem)
#undef QMD_TRACE_HEAD
#defineQMD_TRACE_HEAD(head)
#undef QMD_SAVELINK
#defineQMD_SAVELINK(name, link)
#undef TRACEBUF
#defineTRACEBUF
#undef TRACEBUF_INITIALIZER

```

```

#define TRACEBUF_INITIALIZER
#undef TRASHIT
#define TRASHIT(x)
#endif /* QUEUE_MACRO_DEBUG */

/*
 * Singly-linked List declarations.
 */
#undef SLIST_HEAD
#define SLIST_HEAD(name, type)\
struct name {\
    struct type *slh_first; /* first element */\
}

#undef SLIST_HEAD_INITIALIZER
#define SLIST_HEAD_INITIALIZER(head)\
{ NULL }

#undef SLIST_ENTRY
#define SLIST_ENTRY(type)\
struct {\
    struct type *sle_next; /* next element */\
}

/*
 * Singly-linked List functions.
 */
#undef SLIST_EMPTY
#define SLIST_EMPTY(head)((head)->slh_first == NULL)

#undef SLIST_FIRST
#define SLIST_FIRST(head)((head)->slh_first)

#undef SLIST_FOREACH
#define SLIST_FOREACH(var, head, field)\
for ((var) = SLIST_FIRST((head));\
    (var);\
    (var) = SLIST_NEXT((var), field))

#undef SLIST_FOREACH_FROM
#define SLIST_FOREACH_FROM(var, head, field)\
for ((var) = ((var) ? (var) : SLIST_FIRST((head));\
    (var);\
    (var) = SLIST_NEXT((var), field))

#undef SLIST_FOREACH_SAFE
#define SLIST_FOREACH_SAFE(var, head, field, tvar)\
for ((var) = SLIST_FIRST((head));\
    (var) && ((tvar) = SLIST_NEXT((var), field), 1);\
    (var) = (tvar))

#undef SLIST_FOREACH_FROM_SAFE
#define SLIST_FOREACH_FROM_SAFE(var, head, field, tvar)\
for ((var) = ((var) ? (var) : SLIST_FIRST((head));\
    (var) && ((tvar) = SLIST_NEXT((var), field), 1);\

```

```

    (var) = (tvar))

#undef SLIST_FOREACH_PREVPTR
#define SLIST_FOREACH_PREVPTR(var, varp, head, field)\
for ((varp) = &SLIST_FIRST((head));\
    ((var) = *(varp)) != NULL;\
    (varp) = &SLIST_NEXT((var), field))

#undef SLIST_INIT
#define SLIST_INIT(head) do {\
    SLIST_FIRST((head)) = NULL;\
} while (0)

#undef SLIST_INSERT_AFTER
#define SLIST_INSERT_AFTER(slistelm, elm, field) do {\
    SLIST_NEXT((elm), field) = SLIST_NEXT((slistelm), field);\
    SLIST_NEXT((slistelm), field) = (elm);\
} while (0)

#undef SLIST_INSERT_HEAD
#define SLIST_INSERT_HEAD(head, elm, field) do {\
    SLIST_NEXT((elm), field) = SLIST_FIRST((head));\
    SLIST_FIRST((head)) = (elm);\
} while (0)

#undef SLIST_NEXT
#define SLIST_NEXT(elm, field)((elm)->field.sle_next)

#undef SLIST_REMOVE
#define SLIST_REMOVE(head, elm, type, field) do {\
    QMD_SAVELINK(oldnext, (elm)->field.sle_next);\
    if (SLIST_FIRST((head)) == (elm)) {\
        SLIST_REMOVE_HEAD((head), field);\
    }\
    else {\
        struct type *curelm = SLIST_FIRST((head));\
        while (SLIST_NEXT(curelm, field) != (elm))\
            curelm = SLIST_NEXT(curelm, field);\
        SLIST_REMOVE_AFTER(curelm, field);\
    }\
    TRASHIT(*oldnext);\
} while (0)

#undef SLIST_REMOVE_AFTER
#define SLIST_REMOVE_AFTER(elm, field) do {\
    SLIST_NEXT(elm, field) =\
        SLIST_NEXT(SLIST_NEXT(elm, field), field);\
} while (0)

#undef SLIST_REMOVE_HEAD
#define SLIST_REMOVE_HEAD(head, field) do {\
    SLIST_FIRST((head)) = SLIST_NEXT(SLIST_FIRST((head)), field);\
} while (0)

```

```

#undef SLIST_SWAP
#define SLIST_SWAP(head1, head2, type) do {\
    struct type *swap_first = SLIST_FIRST(head1);\
    SLIST_FIRST(head1) = SLIST_FIRST(head2);\
    SLIST_FIRST(head2) = swap_first;\
} while (0)

/*
 * Singly-linked Tail queue declarations.
 */
#undef STAILQ_HEAD
#define STAILQ_HEAD(name, type)\
    struct name {\
        struct type *stqh_first; /* first element */\
        struct type **stqh_last; /* addr of last next element */\
    }

#undef STAILQ_HEAD_INITIALIZER
#define STAILQ_HEAD_INITIALIZER(head)\
    { NULL, &(head).stqh_first }

#undef STAILQ_ENTRY
#define STAILQ_ENTRY(type)\
    struct {\
        struct type *stqe_next; /* next element */\
    }

/*
 * Singly-linked Tail queue functions.
 */
#undef STAILQ_CONCAT
#define STAILQ_CONCAT(head1, head2) do {\
    if (!STAILQ_EMPTY((head2))) {\
        *(head1)->stqh_last = (head2)->stqh_first;\
        (head1)->stqh_last = (head2)->stqh_last;\
        STAILQ_INIT((head2));\
    }\
} while (0)

#undef STAILQ_EMPTY
#define STAILQ_EMPTY(head)((head)->stqh_first == NULL)

#undef STAILQ_FIRST
#define STAILQ_FIRST(head)((head)->stqh_first)

#undef STAILQ_FOREACH
#define STAILQ_FOREACH(var, head, field)\
    for ((var) = STAILQ_FIRST((head));\
        (var);\
        (var) = STAILQ_NEXT((var), field))

#undef STAILQ_FOREACH_FROM
#define STAILQ_FOREACH_FROM(var, head, field)\
    for ((var) = ((var) ? (var) : STAILQ_FIRST((head)));\
        (var);\

```

```

    (var) = STAILQ_NEXT((var), field))

#undef STAILQ_FOREACH_SAFE
#define STAILQ_FOREACH_SAFE(var, head, field, tvar) \
for ((var) = STAILQ_FIRST((head)); \
    (var) && ((tvar) = STAILQ_NEXT((var), field), 1); \
    (var) = (tvar))

#undef STAILQ_FOREACH_FROM_SAFE
#define STAILQ_FOREACH_FROM_SAFE(var, head, field, tvar) \
for ((var) = ((var) ? (var) : STAILQ_FIRST((head))); \
    (var) && ((tvar) = STAILQ_NEXT((var), field), 1); \
    (var) = (tvar))

#undef STAILQ_INIT
#define STAILQ_INIT(head) do { \
    STAILQ_FIRST((head)) = NULL; \
    (head)->stqh_last = &STAILQ_FIRST((head)); \
} while (0)

#undef STAILQ_INSERT_AFTER
#define STAILQ_INSERT_AFTER(head, tqelm, elm, field) do { \
    if ((STAILQ_NEXT((elm), field) = STAILQ_NEXT((tqelm), field)) == NULL) \
    (head)->stqh_last = &STAILQ_NEXT((elm), field); \
    STAILQ_NEXT((tqelm), field) = (elm); \
} while (0)

#undef STAILQ_INSERT_HEAD
#define STAILQ_INSERT_HEAD(head, elm, field) do { \
    if ((STAILQ_NEXT((elm), field) = STAILQ_FIRST((head))) == NULL) \
    (head)->stqh_last = &STAILQ_NEXT((elm), field); \
    STAILQ_FIRST((head)) = (elm); \
} while (0)

#undef STAILQ_INSERT_TAIL
#define STAILQ_INSERT_TAIL(head, elm, field) do { \
    STAILQ_NEXT((elm), field) = NULL; \
    *(head)->stqh_last = (elm); \
    (head)->stqh_last = &STAILQ_NEXT((elm), field); \
} while (0)

#undef STAILQ_LAST
#define STAILQ_LAST(head, type, field) \
(STAILQ_EMPTY((head)) ? NULL : \
    __containerof((head)->stqh_last, struct type, field.stqe_next))

#undef STAILQ_NEXT
#define STAILQ_NEXT(elm, field)((elm)->field.stqe_next)

#undef STAILQ_REMOVE
#define STAILQ_REMOVE(head, elm, type, field) do { \
    QMD_SAVELINK(oldnext, (elm)->field.stqe_next); \
    if (STAILQ_FIRST((head)) == (elm)) { \
        STAILQ_REMOVE_HEAD((head), field); \

```

```

}\
else {\
struct type *curelm = STAILQ_FIRST((head));\
while (STAILQ_NEXT(curelm, field) != (elm))\
curelm = STAILQ_NEXT(curelm, field);\
STAILQ_REMOVE_AFTER(head, curelm, field);\
}\
TRASHIT(*oldnext);\
} while (0)

#undef STAILQ_REMOVE_AFTER
#define STAILQ_REMOVE_AFTER(head, elm, field) do {\
if ((STAILQ_NEXT(elm, field) =\
    STAILQ_NEXT(STAILQ_NEXT(elm, field), field)) == NULL)\
(head)->stqh_last = &STAILQ_NEXT((elm), field);\
} while (0)

#undef STAILQ_REMOVE_HEAD
#define STAILQ_REMOVE_HEAD(head, field) do {\
if ((STAILQ_FIRST((head)) =\
    STAILQ_NEXT(STAILQ_FIRST((head)), field)) == NULL)\
(head)->stqh_last = &STAILQ_FIRST((head));\
} while (0)

#undef STAILQ_SWAP
#define STAILQ_SWAP(head1, head2, type) do {\
struct type *swap_first = STAILQ_FIRST(head1);\
struct type **swap_last = (head1)->stqh_last;\
STAILQ_FIRST(head1) = STAILQ_FIRST(head2);\
(head1)->stqh_last = (head2)->stqh_last;\
STAILQ_FIRST(head2) = swap_first;\
(head2)->stqh_last = swap_last;\
if (STAILQ_EMPTY(head1))\
(head1)->stqh_last = &STAILQ_FIRST(head1);\
if (STAILQ_EMPTY(head2))\
(head2)->stqh_last = &STAILQ_FIRST(head2);\
} while (0)

/*
 * List declarations.
 */
#undef LIST_HEAD
#define LIST_HEAD(name, type)\
struct name {\
struct type *lh_first; /* first element */\
}

#undef LIST_HEAD_INITIALIZER
#define LIST_HEAD_INITIALIZER(head)\
{ NULL }

#undef LIST_ENTRY
#define LIST_ENTRY(type)\
struct {\

```

```

struct type *le_next; /* next element */
struct type **le_prev; /* address of previous next element */
}

/*
 * List functions.
 */
#undef LIST_EMPTY
#define LIST_EMPTY(head)((head)->lh_first == NULL)

#undef LIST_FIRST
#define LIST_FIRST(head)((head)->lh_first)

#undef LIST_FOREACH
#define LIST_FOREACH(var, head, field)\
for ((var) = LIST_FIRST((head));\
    (var);\
    (var) = LIST_NEXT((var), field))

#undef LIST_FOREACH_FROM
#define LIST_FOREACH_FROM(var, head, field)\
for ((var) = ((var) ? (var) : LIST_FIRST((head)));\
    (var);\
    (var) = LIST_NEXT((var), field))

#undef LIST_FOREACH_SAFE
#define LIST_FOREACH_SAFE(var, head, field, tvar)\
for ((var) = LIST_FIRST((head));\
    (var) && ((tvar) = LIST_NEXT((var), field), 1);\
    (var) = (tvar))

#undef LIST_FOREACH_FROM_SAFE
#define LIST_FOREACH_FROM_SAFE(var, head, field, tvar)\
for ((var) = ((var) ? (var) : LIST_FIRST((head)));\
    (var) && ((tvar) = LIST_NEXT((var), field), 1);\
    (var) = (tvar))

#undef LIST_INIT
#define LIST_INIT(head) do {\
LIST_FIRST((head)) = NULL;\
} while (0)

#undef LIST_INSERT_AFTER
#define LIST_INSERT_AFTER(listelm, elm, field) do {\
if ((LIST_NEXT((elm), field) = LIST_NEXT((listelm), field)) != NULL)\
LIST_NEXT((listelm), field)->field.le_prev =\
    &LIST_NEXT((elm), field);\
LIST_NEXT((listelm), field) = (elm);\
(elm)->field.le_prev = &LIST_NEXT((listelm), field);\
} while (0)

#undef LIST_INSERT_BEFORE
#define LIST_INSERT_BEFORE(listelm, elm, field) do {\
(elm)->field.le_prev = (listelm)->field.le_prev;\

```

```

LIST_NEXT((elm), field) = (listelm);\
*(listelm)->field.le_prev = (elm);\
(listelm)->field.le_prev = &LIST_NEXT((elm), field);\
} while (0)

#undef LIST_INSERT_HEAD
#define LIST_INSERT_HEAD(head, elm, field) do {\
if ((LIST_NEXT((elm), field) = LIST_FIRST((head))) != NULL)\
LIST_FIRST((head))->field.le_prev = &LIST_NEXT((elm), field);\
LIST_FIRST((head)) = (elm);\
(elm)->field.le_prev = &LIST_FIRST((head));\
} while (0)

#undef LIST_NEXT
#define LIST_NEXT(elm, field)((elm)->field.le_next)

#undef LIST_PREV
#define LIST_PREV(elm, head, type, field)\
((elm)->field.le_prev == &LIST_FIRST((head)) ? NULL :\
__containerof((elm)->field.le_prev, struct type, field.le_next))

#undef LIST_REMOVE
#define LIST_REMOVE(elm, field) do {\
QMD_SAVELINK(oldnext, (elm)->field.le_next);\
QMD_SAVELINK(oldprev, (elm)->field.le_prev);\
if (LIST_NEXT((elm), field) != NULL)\
LIST_NEXT((elm), field)->field.le_prev = \
(elm)->field.le_prev;\
*(elm)->field.le_prev = LIST_NEXT((elm), field);\
TRASHIT(*oldnext);\
TRASHIT(*oldprev);\
} while (0)

#undef LIST_SWAP
#define LIST_SWAP(head1, head2, type, field) do {\
struct type *swap_tmp = LIST_FIRST((head1));\
LIST_FIRST((head1)) = LIST_FIRST((head2));\
LIST_FIRST((head2)) = swap_tmp;\
if ((swap_tmp = LIST_FIRST((head1))) != NULL)\
swap_tmp->field.le_prev = &LIST_FIRST((head1));\
if ((swap_tmp = LIST_FIRST((head2))) != NULL)\
swap_tmp->field.le_prev = &LIST_FIRST((head2));\
} while (0)

/*
 * Tail queue declarations.
 */
#undef TAILQ_HEAD
#define TAILQ_HEAD(name, type)\
struct name {\
struct type *tqh_first; /* first element */\
struct type **tqh_last; /* addr of last next element */\
TRACEBUF\
}

```



```

#undef TAILQ_HEAD_INITIALIZER
#define TAILQ_HEAD_INITIALIZER(head)\
{ NULL, &(head).tqh_first, TRACEBUF_INITIALIZER }

#undef TAILQ_ENTRY
#define TAILQ_ENTRY(type)\
struct {\
    struct type *tqe_next; /* next element */\
    struct type **tqe_prev; /* address of previous next element */\
    TRACEBUF\
}

/*
 * Tail queue functions.
 */
#undef TAILQ_CONCAT
#define TAILQ_CONCAT(head1, head2, field) do {\
    if (!TAILQ_EMPTY(head2)) {\
        *(head1)->tqh_last = (head2)->tqh_first;\
        (head2)->tqh_first->field.tqe_prev = (head1)->tqh_last;\
        (head1)->tqh_last = (head2)->tqh_last;\
        TAILQ_INIT((head2));\
        QMD_TRACE_HEAD(head1);\
        QMD_TRACE_HEAD(head2);\
    }\
} while (0)

#undef TAILQ_EMPTY
#define TAILQ_EMPTY(head)((head)->tqh_first == NULL)

#undef TAILQ_FIRST
#define TAILQ_FIRST(head)((head)->tqh_first)

#undef TAILQ_FOREACH
#define TAILQ_FOREACH(var, head, field)\
for ((var) = TAILQ_FIRST((head));\
    (var);\
    (var) = TAILQ_NEXT((var), field))

#undef TAILQ_FOREACH_FROM
#define TAILQ_FOREACH_FROM(var, head, field)\
for ((var) = ((var) ? (var) : TAILQ_FIRST((head)));\
    (var);\
    (var) = TAILQ_NEXT((var), field))

#undef TAILQ_FOREACH_SAFE
#define TAILQ_FOREACH_SAFE(var, head, field, tvar)\
for ((var) = TAILQ_FIRST((head));\
    (var) && ((tvar) = TAILQ_NEXT((var), field), 1);\
    (var) = (tvar))

#undef TAILQ_FOREACH_FROM_SAFE
#define TAILQ_FOREACH_FROM_SAFE(var, head, field, tvar)\
for ((var) = ((var) ? (var) : TAILQ_FIRST((head)));\

```

```

        (var) && ((tvar) = TAILQ_NEXT((var), field), 1);\
        (var) = (tvar))

#undef TAILQ_FOREACH_REVERSE
#defineTAILQ_FOREACH_REVERSE(var, head, headname, field)\
for ((var) = TAILQ_LAST((head), headname);\
    (var);\
    (var) = TAILQ_PREV((var), headname, field))

#undef TAILQ_FOREACH_REVERSE_FROM
#defineTAILQ_FOREACH_REVERSE_FROM(var, head, headname, field)\
for ((var) = ((var) ? (var) : TAILQ_LAST((head), headname));\
    (var);\
    (var) = TAILQ_PREV((var), headname, field))

#undef TAILQ_FOREACH_REVERSE_SAFE
#defineTAILQ_FOREACH_REVERSE_SAFE(var, head, headname, field, tvar)\
for ((var) = TAILQ_LAST((head), headname);\
    (var) && ((tvar) = TAILQ_PREV((var), headname, field), 1);\
    (var) = (tvar))

#undef TAILQ_FOREACH_REVERSE_FROM_SAFE
#defineTAILQ_FOREACH_REVERSE_FROM_SAFE(var, head, headname, field, tvar) \
for ((var) = ((var) ? (var) : TAILQ_LAST((head), headname));\
    (var) && ((tvar) = TAILQ_PREV((var), headname, field), 1);\
    (var) = (tvar))

#undef TAILQ_INIT
#defineTAILQ_INIT(head) do {\
    TAILQ_FIRST((head)) = NULL;\
    (head)->tqh_last = &TAILQ_FIRST((head));\
    QMD_TRACE_HEAD(head);\
} while (0)

#undef TAILQ_INSERT_AFTER
#defineTAILQ_INSERT_AFTER(head, listelm, elm, field) do {\
    if ((TAILQ_NEXT((elm), field) = TAILQ_NEXT((listelm), field)) != NULL)\
        TAILQ_NEXT((elm), field)->field.tqe_prev = \
            &TAILQ_NEXT((elm), field);\
    else {\
        (head)->tqh_last = &TAILQ_NEXT((elm), field);\
        QMD_TRACE_HEAD(head);\
    }\
    TAILQ_NEXT((listelm), field) = (elm);\
    (elm)->field.tqe_prev = &TAILQ_NEXT((listelm), field);\
    QMD_TRACE_ELEM(&(elm)->field);\
    QMD_TRACE_ELEM(&listelm->field);\
} while (0)

#undef TAILQ_INSERT_BEFORE
#defineTAILQ_INSERT_BEFORE(listelm, elm, field) do {\
    (elm)->field.tqe_prev = (listelm)->field.tqe_prev;\
    TAILQ_NEXT((elm), field) = (listelm);\
    *(listelm)->field.tqe_prev = (elm);\
    (listelm)->field.tqe_prev = &TAILQ_NEXT((elm), field);\
} while (0)

```

```

QMD_TRACE_ELEM(&(elm)->field);\
QMD_TRACE_ELEM(&listelm->field);\
} while (0)

#undef TAILQ_INSERT_HEAD
#define TAILQ_INSERT_HEAD(head, elm, field) do {\
if ((TAILQ_NEXT((elm), field) = TAILQ_FIRST((head))) != NULL)\
TAILQ_FIRST((head))->field.tqe_prev =\
    &TAILQ_NEXT((elm), field);\
else\
(head)->tqh_last = &TAILQ_NEXT((elm), field);\
TAILQ_FIRST((head)) = (elm);\
(elm)->field.tqe_prev = &TAILQ_FIRST((head));\
QMD_TRACE_HEAD(head);\
QMD_TRACE_ELEM(&(elm)->field);\
} while (0)

#undef TAILQ_INSERT_TAIL
#define TAILQ_INSERT_TAIL(head, elm, field) do {\
TAILQ_NEXT((elm), field) = NULL;\
(elm)->field.tqe_prev = (head)->tqh_last;\
*(head)->tqh_last = (elm);\
(head)->tqh_last = &TAILQ_NEXT((elm), field);\
QMD_TRACE_HEAD(head);\
QMD_TRACE_ELEM(&(elm)->field);\
} while (0)

#undef TAILQ_LAST
#define TAILQ_LAST(head, headname)\
(*(((struct headname *)((head)->tqh_last))->tqh_last))

#undef TAILQ_NEXT
#define TAILQ_NEXT(elm, field) ((elm)->field.tqe_next)

#undef TAILQ_PREV
#define TAILQ_PREV(elm, headname, field)\
(*(((struct headname *)((elm)->field.tqe_prev))->tqh_last))

#undef TAILQ_REMOVE
#define TAILQ_REMOVE(head, elm, field) do {\
QMD_SAVELINK(oldnext, (elm)->field.tqe_next);\
QMD_SAVELINK(oldprev, (elm)->field.tqe_prev);\
if ((TAILQ_NEXT((elm), field)) != NULL)\
TAILQ_NEXT((elm), field)->field.tqe_prev = \
    (elm)->field.tqe_prev;\
else {\
(head)->tqh_last = (elm)->field.tqe_prev;\
QMD_TRACE_HEAD(head);\
}\
*(elm)->field.tqe_prev = TAILQ_NEXT((elm), field);\
TRASHIT(*oldnext);\
TRASHIT(*oldprev);\
QMD_TRACE_ELEM(&(elm)->field);\
} while (0)

```

```

#undef TAILQ_SWAP
#define TAILQ_SWAP(head1, head2, type, field) do {\
    struct type *swap_first = (head1)->tqh_first;\
    struct type **swap_last = (head1)->tqh_last;\
    (head1)->tqh_first = (head2)->tqh_first;\
    (head1)->tqh_last = (head2)->tqh_last;\
    (head2)->tqh_first = swap_first;\
    (head2)->tqh_last = swap_last;\
    if ((swap_first = (head1)->tqh_first) != NULL)\
        swap_first->field.tqe_prev = &(head1)->tqh_first;\
    else\
        (head1)->tqh_last = &(head1)->tqh_first;\
    if ((swap_first = (head2)->tqh_first) != NULL)\
        swap_first->field.tqe_prev = &(head2)->tqh_first;\
    else\
        (head2)->tqh_last = &(head2)->tqh_first;\
} while (0)

/*
 * Headless Tail queue definitions.
 */
#undef HLTQ_ENTRY
#define HLTQ_ENTRY(type)TAILQ_ENTRY(type)

#undef HLTQ_INIT
#define HLTQ_INIT(entry, field) do {\
    (entry)->field.tqe_next = NULL;\
    (entry)->field.tqe_prev = &(entry)->field.tqe_next;\
} while (0)

#undef HLTQ_INITIALIZER
#define HLTQ_INITIALIZER(entry, field)\
    { NULL, &(entry)->field.tqe_next }

#undef HLTQ_FIRST
#define HLTQ_FIRST(elm)(elm)

#undef HLTQ_END
#define HLTQ_END(elm)NULL

#undef HLTQ_NEXT
#define HLTQ_NEXT(elm, field)((elm)->field.tqe_next)

#undef HLTQ_LAST
#define HLTQ_LAST(elm, type, field)\
    ((elm)->field.tqe_next == NULL ? (elm) :\
    __containerof((elm)->field.tqe_prev, struct type, field.tqe_next))

#undef HLTQ_PREV
#define HLTQ_PREV(elm, type, field)\
    (*(elm)->field.tqe_prev == NULL ? NULL :\
    __containerof((elm)->field.tqe_prev, struct type, field.tqe_next))

#undef HLTQ_FOREACH

```

```

#define HLTQ_FOREACH(var, head, field)\
for ((var) = HLTQ_FIRST(head);\
    (var) != HLTQ_END(head);\
    (var) = HLTQ_NEXT(var, field))

#undef HLTQ_FOREACH_SAFE
#define HLTQ_FOREACH_SAFE(var, head, field, tvar)\
for ((var) = HLTQ_FIRST(head);\
    (var) != HLTQ_END(head) &&\
    ((tvar) = HLTQ_NEXT(var, field), 1);\
    (var) = (tvar))

#undef HLTQ_FOREACH_REVERSE
#define HLTQ_FOREACH_REVERSE(var, head, headname, field)\
for ((var) = HLTQ_LAST(head, headname);\
    (var) != HLTQ_END(head);\
    (var) = HLTQ_PREV(var, headname, field))

#undef HLTQ_FOREACH_REVERSE_SAFE
#define HLTQ_FOREACH_REVERSE_SAFE(var, head, headname, field, tvar)\
for ((var) = HLTQ_LAST(head, headname);\
    (var) != HLTQ_END(head) &&\
    ((tvar) = HLTQ_PREV(var, headname, field), 1);\
    (var) = (tvar))

/* Concatenate queue2 to the end of queue1. */
#undef HLTQ_CONCAT
#define HLTQ_CONCAT(queue1, queue2, field) do {\
    (queue2)->field.tqe_prev = (queue1)->field.tqe_prev;\
    *(queue1)->field.tqe_prev = (queue2);\
    (queue1)->field.tqe_prev = &(queue2)->field.tqe_next;\
} while (0)

/* Convert a headless tailq to a headful one. */
#define HLTQ_TO_TAILQ(head, hl, field) do {\
    (head)->tqh_first = (hl);\
    (head)->tqh_last = (hl)->field.tqe_prev;\
    (hl)->field.tqe_prev = &(head)->tqh_first;\
} while (0)

/* Concatenate a headless tail queue to the end of a regular tail queue. */
#define TAILQ_CONCAT_HLTQ(head, hl, field) do {\
    void *last = (hl)->field.tqe_prev;\
    (hl)->field.tqe_prev = (head)->tqh_last;\
    *(head)->tqh_last = (hl);\
    (head)->tqh_last = last;\
} while (0)

#endif /* !SUDO_QUEUE_H */

```

1.130 syslogd

1.130.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

/*

klogd.c - main program for Linux kernel log daemon.
Copyright (c) 1995 Dr. G.W. Wettstein <greg@wind.rmcc.com>

This file is part of the sysklogd package, a kernel and system log daemon.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

*/

/*

* Steve Lord (lord@cray.com) 7th Nov 92

*

* Modified to check for kernel info by Dr. G.W. Wettstein 02/17/93.

*

* Fri Mar 12 16:53:56 CST 1993: Dr. Wettstein

* Modified LogLine to use a newline as the line separator in
*the kernel message buffer.

*

* Added debugging code to dump the contents of the kernel message
*buffer at the start of the LogLine function.

*

* Thu Jul 29 11:40:32 CDT 1993: Dr. Wettstein

* Added syscalls to turn off logging of kernel messages to the
*console when klogd becomes responsible for kernel messages.

*
*klogd now catches SIGTERM and SIGKILL signals. Receipt of these
*signals cases the clean_up function to be called which shuts down
*kernel logging and re-enables logging of messages to the console.
*
* Sat Dec 11 11:54:22 CST 1993: Dr. Wettstein
*Added fixes to allow compilation with no complaints with -Wall.
*
* When the daemon catches a fatal signal (SIGTERM, SIGKILL) a
*message is output to the logfile advising that the daemon is
*going to terminate.
*
* Thu Jan 6 11:54:10 CST 1994: Dr. Wettstein
*Major re-write/re-organization of the code.
*
*Klogd now assigns kernel messages to priority levels when output
*to the syslog facility is requested. The priority level is
*determined by decoding the prioritization sequence which is
*tagged onto the start of the kernel messages.
*
*Added the following program options: -f arg -c arg -s -o -d
*
*The -f switch can be used to specify that output should
*be written to the named file.
*
*The -c switch is used to specify the level of kernel
*messages which are to be directed to the console.
*
*The -s switch causes the program to use the syscall
*interface to the kernel message facility. This can be
*used to override the presence of the /proc filesystem.
*
*The -o switch causes the program to operate in 'one-shot'
*mode. A single call will be made to read the complete
*kernel buffer. The contents of the buffer will be
*output and the program will terminate.
*
*The -d switch causes 'debug' mode to be activated. This
*will cause the daemon to generate LOTS of output to stderr.
*
*The buffer decomposition function (LogLine) was re-written to
*squash a bug which was causing only partial kernel messages to
*be written to the syslog facility.
*
*The signal handling code was modified to properly differentiate
*between the STOP and TSTP signals.
*
*Added pid saving when the daemon detaches into the background. Thank
*you to Juha Virtanen (jiivee@hut.fi) for providing this patch.
*
* Mon Feb 6 07:31:29 CST 1995: Dr. Wettstein
*Significant re-organization of the signal handling code. The
*signal handlers now only set variables. Not earth shaking by any
*means but aesthetically pleasing to the code purists in the group.

*
*Patch to make things more compliant with the file system standards.
*Thanks to Chris Metcalf for prompting this helpful change.
*
*The routines responsible for reading the kernel log sources now
*initialize the buffers before reading. I think that this will
*solve problems with non-terminated kernel messages producing
*output of the form: new old old old
*
*This may also help influence the occasional reports of klogd
*failing under significant load. I think that the jury may still
*be out on this one though. My thanks to Joerg Ahrens for initially
*tipping me off to the source of this problem. Also thanks to
*Michael O'Reilly for tipping me off to the best fix for this problem.
*And last but not least Mark Lord for prompting me to try this as
*a means of attacking the stability problem.
*
*Specifying a - as the argument to the -f switch will cause output
*to be directed to stdout rather than a filename of -. Thanks to
*Randy Appleton for a patch which prompted me to do this.
*
* Wed Feb 22 15:37:37 CST 1995: Dr. Wettstein
*Added version information to logging startup messages.
*
* Wed Jul 26 18:57:23 MET DST 1995: Martin Schulze
*Added an commandline argument "-n" to avoid forking. This obsoletes
*the compiler define NO_FORK. It's more useful to have this as an
*argument as there are many binary versions and one doesn't need to
*recompile the daemon.
*
* Thu Aug 10 19:01:08 MET DST 1995: Martin Schulze
*Added my pidfile.[ch] to it to perform a better handling with pidfiles.
*Now both, syslogd and klogd, can only be started once. They check the
*pidfile.
*
* Fri Nov 17 15:05:43 CST 1995: Dr. Wettstein
*Added support for kernel address translation. This required moving
*some definitions and includes to the new klogd.h file. Some small
*code cleanups and modifications.
*
* Mon Nov 20 10:03:39 MET 1995
*Added -v option to print the version and exit.
*
* Thu Jan 18 11:19:46 CST 1996: Dr. Wettstein
*Added suggested patches from beta-testers. These address two
*two problems. The first is segmentation faults which occur with
*the ELF libraries. This was caused by passing a null pointer to
*the strcmp function.
*
*Added a second patch to remove the pidfile as part of the
*termination cleanup sequence. This minimizes the potential for
*conflicting pidfiles causing immediate termination at boot time.
*
* Wed Aug 21 09:13:03 CDT 1996: Dr. Wettstein
*Added ability to reload static symbols and kernel module symbols

* under control of SIGUSR1 and SIGUSR2 signals.
*
* Added -p switch to select 'paranoid' behavior with respect to the
* loading of kernel module symbols.
*
* Informative line now printed whenever a state change occurs due
* to signal reception by the daemon.
*
* Added the -i and -I command line switches to signal the currently
* executing daemon.
*
* Tue Nov 19 10:15:36 PST 1996: Leland Olds <olds@eskimo.com>
* Corrected vulnerability to buffer overruns by rewriting LogLine
* routine. Obscenely long kernel messages will now be broken up
* into lines no longer than LOG_LINE_LENGTH.
*
* The last version of LogLine was vulnerable to buffer overruns:
* - Kernel messages longer than LOG_LINE_LENGTH caused a buffer
* overrun.
* - If a line was determined to be shorter than LOG_LINE_LENGTH,
* the routine "ExpandKadds" could cause the line grow by
* an unknown amount and overrun a buffer.
* I turned these routines into a little parsing state machine that
* should not have these problems.
*
* Sun Jun 15 16:23:29 MET DST 1997: Michael Alan Dorman
* Some more glibc patches made by <mdorman@debian.org>.
*
* Thu Aug 21 12:11:27 MET DST 1997: Martin Schulze <joe@infodrom.north.de>
* Fixed little mistake which prevented klogd from accepting a
* console log
*
* Fri Jan 9 00:39:52 CET 1998: Martin Schulze <joe@infodrom.north.de>
* Changed the behaviour of klogd when receiving a terminate
* signal. Now the program terminates immediately instead of
* completing the receipt of a kernel message, i.e the read()
* call. The old behaviour could result in klogd being
* recognized as being undead, because it'll only die after a
* message has been received.
*
* Fri Jan 9 11:03:48 CET 1998: Martin Schulze <joe@infodrom.north.de>
* Corrected some code that caused klogd to dump core when
* receiving messages containing '%', some of them exist in
* 2.1.78. Thanks to Chu-yeon Park <kokids@doit.ajou.ac.kr> for
* informing me.
*
* Fri Jan 9 23:38:19 CET 1998: Florian La Roche <florian@knorke.saar.de>
* Added -x switch to omit EIP translation and System.map evaluation.
*
* Sun Jan 25 20:47:46 CET 1998: Martin Schulze <joe@infodrom.north.de>
* As the bug covering the %'s introduced a problem with
* unevaluated priorities I've worked out a real fix that strips
* %'s to an even number which is harmless for printf.
*

* Sat Oct 10 20:01:48 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Added support for TESTING define which will turn klogd into
*stdio-mode used for debugging.
*
* Mon Apr 13 18:18:45 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Modified System.map read function to try all possible map
*files until a file with matching version is found. Added support for
*Debian release.
*
* Mon Oct 12 13:01:27 MET DST 1998: Martin Schulze <joey@infodrom.north.de>
*Used unsigned long and strtoul() to resolve kernel oops symbols.
*
* Sun Jan 3 18:38:03 CET 1999: Martin Schulze <joey@infodrom.north.de>
*Shortened LOG_LINE_LENGTH in order to get long lines splitted
*up earlier and syslogd has a better chance concatenating them
*together again.
*
* Sat Aug 21 12:27:02 CEST 1999: Martin Schulze <joey@infodrom.north.de>
*Skip newline when reading in messages.
*
* Tue Sep 12 22:14:33 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>
*Don't feed a buffer directly to a printf-type routine, use
*"%s" as format string instead. Thanks to Jouko Pynnönen
*<jouko@solutions.fi> for pointing this out.
*
* Tue Sep 12 22:44:57 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>
* Commandline option '-2': When symbols are expanded, print the
*line twice. Once with addresses converted to symbols, once with the
*raw text. Allows external programs such as ksymoops do their own
*processing on the original data. Thanks to Keith Owens
*<kaos@ocs.com.au> for the patch.
*
* Mon Sep 18 09:32:27 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>
*Added patch to fix priority decoding after moving kernel
*messages into "%s". Thanks to Solar Designer
*<solar@false.com> for the patch.
*
* Sun Mar 11 20:23:44 CET 2001: Martin Schulze <joey@infodrom.ffis.de>
*Stop LogLine() from being called with wrong argument when a
*former calculation failed already. Thanks to Thomas Roessler
*<roessler@does-not-exist.org> for providing a patch.
*
*Ignore zero bytes, no busy loop is entered anymore. Several
*people have submitted patches: Troels Walsted Hansen
*<troels@thule.no>, Wolfgang Oertl <Wolfgang.Oertl@uibk.ac.at>
*and Thomas Roessler.
*
* Thu Apr 29 15:24:07 2004: Solar Designer <solar@openwall.com>
*Prevent potential buffer overflow in reading messages from the
*kernel log rinbuffer.
*
* Sat May 26 16:33:18 2007: Martin Schulze <joey@infodrom.org>
*Improved daemonise routine to stabilise startup.
*
* Mon May 28 18:07:59 CEST 2007: Matthew Fischer <futhark@vzavenue.net>


```
*Remove special treatment of the percent sign.
*/

/* Includes. */
#include <unistd.h>
#include <signal.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <stdarg.h>
#include <paths.h>
#include <stdlib.h>
#include "klogd.h"
#include "ksyms.h"
#ifdef TESTING
#include "pidfile.h"
#endif
#include "version.h"

#define __LIBRARY__
#include <linux/unistd.h>
#include <sys/klog.h>
#define ksyslog klogctl

#define LOG_BUFFER_SIZE 4096
#define LOG_LINE_LENGTH 1000

#ifdef TESTING
#ifdef FSSTND
static char*PidFile = _PATH_VARRUN "klogd.pid";
#else
static char*PidFile = "/etc/klogd.pid";
#endif
#endif

static intkmsg,
change_state = 0,
terminate = 0,
caught_TSTP = 0,
reload_symbols = 0,
console_log_level = -1;

static intuse_syscall = 0,
one_shot = 0,
symbol_lookup = 1,
no_fork = 0; /* don't fork - don't run in daemon mode */

static char*symfile = (char *) 0,
log_buffer[LOG_BUFFER_SIZE];

static FILE *output_file = (FILE *) 0;

static enum LOGSRC {none, proc, kernel} logsrc;
```

```

int debugging = 0;
int symbols_twice = 0;

/* Function prototypes. */
extern int ksyslog(int type, char *buf, int len);
static void CloseLogSrc(void);
extern void restart(int sig);
extern void stop_logging(int sig);
extern void stop_daemon(int sig);
extern void reload_daemon(int sig);
static void Terminate(void);
static void SignalDaemon(int);
static void ReloadSymbols(void);
static void ChangeLogging(void);
static enum LOGSRC GetKernelLogSrc(void);
static void LogLine(char *ptr, int len);
static void LogKernelLine(void);
static void LogProcLine(void);
extern int main(int argc, char *argv[]);

static void CloseLogSrc()

{
    /* Shutdown the log sources. */
    switch ( logsrc )
    {
        case kernel:
            ksyslog(0, 0, 0);
            Syslog(LOG_INFO, "Kernel logging (ksyslog) stopped.");
            break;
        case proc:
            close(kmsg);
            Syslog(LOG_INFO, "Kernel logging (proc) stopped.");
            break;
        case none:
            break;
    }

    if ( output_file != (FILE *) 0 )
        fflush(output_file);
    return;
}

/*
 * Signal handler to terminate the parent process.
 */
#ifdef TESTING
void doexit(sig)

int sig;

```

```
{
exit (0);
}
#endif

void restart(sig)

int sig;

{
signal(SIGCONT, restart);
change_state = 1;
caught_TSTP = 0;
return;
}

void stop_logging(sig)

int sig;

{
signal(SIGTSTP, stop_logging);
change_state = 1;
caught_TSTP = 1;
return;
}

void stop_daemon(sig)

int sig;

{
Terminate();
return;
}

void reload_daemon(sig)

int sig;

{
change_state = 1;
reload_symbols = 1;

if ( sig == SIGUSR2 )
{
++reload_symbols;
signal(SIGUSR2, reload_daemon);
}
else
```

```
signal(SIGUSR1, reload_daemon);

return;
}

static void Terminate()
{
    CloseLogSrc();
    Syslog(LOG_INFO, "Kernel log daemon terminating.");
    sleep(1);
    if ( output_file != (FILE *) 0 )
        fclose(output_file);
    closelog();
#ifdef TESTING
    (void) remove_pid(PidFile);
#endif
    exit(1);
}

static void SignalDaemon(sig)

    int sig;

{
#ifdef TESTING
    auto int pid = check_pid(PidFile);

    kill(pid, sig);
#else
    kill(getpid(), sig);
#endif
    return;
}

static void ReloadSymbols()

{
    if (symbol_lookup) {
        if ( reload_symbols > 1 )
            InitKsyms(symfile);
        InitMsyms();
    }
    reload_symbols = change_state = 0;
    return;
}

static void ChangeLogging(void)

{
    /* Terminate kernel logging. */
    if ( terminate == 1 )
```

```
Terminate();

/* Indicate that something is happening. */
Syslog(LOG_INFO, "klogd %s.%s, ----- state change -----\n", \
        VERSION, PATCHLEVEL);

/* Reload symbols. */
if ( reload_symbols > 0 )
{
    ReloadSymbols();
    return;
}

/* Stop kernel logging. */
if ( caught_TSTP == 1 )
{
    CloseLogSrc();
    logsrc = none;
    change_state = 0;
    return;
}

/*
 * The rest of this function is responsible for restarting
 * kernel logging after it was stopped.
 *
 * In the following section we make a decision based on the
 * kernel log state as to what is causing us to restart. Somewhat
 * groady but it keeps us from creating another static variable.
 */
if ( logsrc != none )
{
    Syslog(LOG_INFO, "Kernel logging re-started after SIGSTOP.");
    change_state = 0;
    return;
}

/* Restart logging. */
logsrc = GetKernelLogSrc();
change_state = 0;
return;
}

static enum LOGSRC GetKernelLogSrc(void)
{
    auto struct stat sb;

    /* Set level of kernel console messaging.. */
    if ( (console_log_level != -1)
        && (ksyslog(8, NULL, console_log_level) < 0) && \
            (errno == EINVAL) )
```

```

{
/*
 * An invalid argument error probably indicates that
 * a pre-0.14 kernel is being run. At this point we
 * issue an error message and simply shut-off console
 * logging completely.
 */
Syslog(LOG_WARNING, "Cannot set console log level - disabling "
        "console output.");
}

/*
 * First do a stat to determine whether or not the proc based
 * file system is available to get kernel messages from.
 */
if ( use_syscall ||
    ((stat(_PATH_KLOG, &sb) < 0) && (errno == ENOENT)) )
{
    /* Initialize kernel logging. */
    ksyslog(1, NULL, 0);
#ifdef DEBRELEASE
    Syslog(LOG_INFO, "klogd %s.%s#%s, log source = ksyslog "
            "started.", VERSION, PATCHLEVEL, DEBRELEASE);
#else
    Syslog(LOG_INFO, "klogd %s.%s, log source = ksyslog "
            "started.", VERSION, PATCHLEVEL);
#endif
    return(kernel);
}

#ifdef TESTING
if ( (kmsg = open(_PATH_KLOG, O_RDONLY)) < 0 )
{
    fprintf(stderr, "klogd: Cannot open proc file system, " \
            "%d - %s.\n", errno, strerror(errno));
    ksyslog(7, NULL, 0);
    exit(1);
}
#else
kmsg = fileno(stdin);
#endif

#ifdef DEBRELEASE
Syslog(LOG_INFO, "klogd %s.%s#%s, log source = %s started.", \
        VERSION, PATCHLEVEL, DEBRELEASE, _PATH_KLOG);
#else
Syslog(LOG_INFO, "klogd %s.%s, log source = %s started.", \
        VERSION, PATCHLEVEL, _PATH_KLOG);
#endif
return(proc);
}

extern void Syslog(int priority, char *fmt, ...)

```

```
{
va_list ap;
char *argl;

if ( debugging )
{
fputs("Logging line:\n", stderr);
fprintf(stderr, "\tLine: %s\n", fmt);
fprintf(stderr, "\tPriority: %d\n", priority);
}

/* Handle output to a file. */
if ( output_file != (FILE *) 0 )
{
va_start(ap, fmt);
vfprintf(output_file, fmt, ap);
va_end(ap);
fputc('\n', output_file);
fflush(output_file);
if (!one_shot)
fsync(fileno(output_file));
return;
}

/* Output using syslog. */
if (!strcmp(fmt, "%s"))
{
va_start(ap, fmt);
argl = va_arg(ap, char *);
if (argl[0] == '<' && argl[1] && argl[2] == '>')
{
switch ( argl[1] )
{
case '0':
priority = LOG_EMERG;
break;
case '1':
priority = LOG_ALERT;
break;
case '2':
priority = LOG_CRIT;
break;
case '3':
priority = LOG_ERR;
break;
case '4':
priority = LOG_WARNING;
break;
case '5':
priority = LOG_NOTICE;
break;
case '6':
priority = LOG_INFO;
break;
```

```

    case '7':
    default:
    priority = LOG_DEBUG;
    }
    argl += 3;
    }
    syslog(priority, fmt, argl);
    va_end(ap);
#ifdef TESTING
    putchar('\n');
#endif
    return;
}

va_start(ap, fmt);
vsyslog(priority, fmt, ap);
va_end(ap);
#ifdef TESTING
    printf("\n");
#endif

return;
}

/*
 *   Copy characters from ptr to line until a char in the delim
 *   string is encountered or until min( space, len ) chars have
 *   been copied.
 *
 *   Returns the actual number of chars copied.
 */
static int copyin( char *line,    int space,
                  const char *ptr, int len,
                  const char *delim )
{
    auto int i;
    auto int count;

    count = len < space ? len : space;

    for(i=0; i<count && !strchr(delim, *ptr); i++) { *line++ = *ptr++; }

    return( i );
}

/*
 * Messages are separated by "\n". Messages longer than
 * LOG_LINE_LENGTH are broken up.
 *
 * Kernel symbols show up in the input buffer as : "<aaaaaa>",
 * where "aaaaaa" is the address. These are replaced with
 * "[symbolname+offset/size]" in the output line - symbolname,
 * offset, and size come from the kernel symbol table.
 *

```



```

* If a kernel symbol happens to fall at the end of a message close
* in length to LOG_LINE_LENGTH, the symbol will not be expanded.
* (This should never happen, since the kernel should never generate
* messages that long.
*
* To preserve the original addresses, lines containing kernel symbols
* are output twice. Once with the symbols converted and again with the
* original text. Just in case somebody wants to run their own Oops
* analysis on the syslog, e.g. ksymoops.
*/
static void LogLine(char *ptr, int len)
{
    enum parse_state_enum {
        PARSING_TEXT,
        PARSING_SYMSTART,    /* at < */
        PARSING_SYMBOL,
        PARSING_SYMEND      /* at ] */
    };

    static char line_buff[LOG_LINE_LENGTH];

    static char *line        =line_buff;
    static enum parse_state_enum parse_state = PARSING_TEXT;
    static int space         = sizeof(line_buff)-1;

    static char *sym_start;    /* points at the '<' of a symbol */

    auto int delta = 0;        /* number of chars copied */
    auto int symbols_expanded = 0; /* 1 if symbols were expanded */
    auto int skip_symbol_lookup = 0; /* skip symbol lookup on this pass */
    auto char *save_ptr = ptr; /* save start of input line */
    auto int save_len = len; /* save length at start of input line */

    while( len > 0 )
    {
        if( space == 0 ) /* line buffer is full */
        {
            /*
             ** Line too long. Start a new line.
             */
            *line = 0; /* force null terminator */

            if ( debugging )
            {
                fputs("Line buffer full:\n", stderr);
                fprintf(stderr, "\tLine: %s\n", line);
            }

            Syslog( LOG_INFO, "%s", line_buff );
            line = line_buff;
            space = sizeof(line_buff)-1;
            parse_state = PARSING_TEXT;
            symbols_expanded = 0;
            skip_symbol_lookup = 0;

```

```

    save_ptr = ptr;
    save_len = len;
}

switch( parse_state )
{
case PARSING_TEXT:
    delta = copyin( line, space, ptr, len, "\n" );
    line += delta;
    ptr += delta;
    space -= delta;
    len -= delta;

    if( space == 0 || len == 0 )
    {
break; /* full line_buff or end of input buffer */
    }

    if( *ptr == '\0' ) /* zero byte */
    {
        ptr++; /* skip zero byte */
        space -= 1;
        len -= 1;
    }

break;
}

    if( *ptr == '\n' ) /* newline */
    {
        ptr++; /* skip newline */
        space -= 1;
        len -= 1;

        *line = 0; /* force null terminator */
        Syslog( LOG_INFO, "%s", line_buff );
        line = line_buff;
        space = sizeof(line_buff)-1;
    }
    if( symbols_twice ) {
        if( symbols_expanded ) {
            /* reprint this line without symbol lookup */
            symbols_expanded = 0;
            skip_symbol_lookup = 1;
            ptr = save_ptr;
            len = save_len;
        }
        else
        {
            skip_symbol_lookup = 0;
            save_ptr = ptr;
            save_len = len;
        }
    }

    break;
}

    if( *ptr == '[' ) /* possible kernel symbol */

```

```

    {
        *line++ = *ptr++;
        space -= 1;
        len -= 1;
    if (!skip_symbol_lookup)
        parse_state = PARSING_SYMSTART;    /* at < */
        break;
    }
    break;

case PARSING_SYMSTART:
    if( *ptr != '<' )
    {
        parse_state = PARSING_TEXT;    /* not a symbol */
        break;
    }

    /*
    ** Save this character for now. If this turns out to
    ** be a valid symbol, this char will be replaced later.
    ** If not, we'll just leave it there.
    */

    sym_start = line; /* this will point at the '<' */

    *line++ = *ptr++;
    space -= 1;
    len -= 1;
    parse_state = PARSING_SYMBOL;    /* symbol... */
    break;

case PARSING_SYMBOL:
    delta = copyin( line, space, ptr, len, ">\n[" );
    line += delta;
    ptr += delta;
    space -= delta;
    len -= delta;
    if( space == 0 || len == 0 )
    {
        break; /* full line_buff or end of input buffer */
    }
    if( *ptr != '>' )
    {
        parse_state = PARSING_TEXT;
        break;
    }

    *line++ = *ptr++; /* copy the '>' */
    space -= 1;
    len -= 1;

    parse_state = PARSING_SYMEND;

    break;

```

```

case PARSING_SYMEND:
    if( *ptr != ']' )
    {
        parse_state = PARSING_TEXT;    /* not a symbol */
        break;
    }

    /*
    ** It's really a symbol! Replace address with the
    ** symbol text.
    */
    {
auto int sym_space;

unsigned long value;
auto struct symbol sym;
auto char *symbol;

        *(line-1) = 0; /* null terminate the address string */
        value = strtoul(sym_start+1, (char **) 0, 16);
        *(line-1) = '>'; /* put back delim */

        symbol = LookupSymbol(value, &sym);
        if ( !symbol_lookup || symbol == (char *) 0 )
        {
            parse_state = PARSING_TEXT;
            break;
        }

        /*
        ** verify there is room in the line buffer
        */
        sym_space = space + ( line - sym_start );
        if( sym_space < strlen(symbol) + 30 ) /*(30 should be overkill)*/
        {
            parse_state = PARSING_TEXT; /* not enough space */
            break;
        }

        delta = sprintf( sym_start, "%s+0x%x/0x%02x]",
                        symbol, sym.offset, sym.size );

        space = sym_space + delta;
        line = sym_start + delta;
symbols_expanded = 1;
    }
    ptr++;
    len--;
    parse_state = PARSING_TEXT;
    break;

default: /* Can't get here! */
    parse_state = PARSING_TEXT;

```

```
    }
}

return;
}

static void LogKernelLine(void)

{
    auto int rdcnt;

    /*
     * Zero-fill the log buffer. This should cure a multitude of
     * problems with klogd logging the tail end of the message buffer
     * which will contain old messages. Then read the kernel log
     * messages into this fresh buffer.
     */
    memset(log_buffer, '\0', sizeof(log_buffer));
    if ( (rdcnt = ksyslog(2, log_buffer, sizeof(log_buffer)-1)) < 0 )
    {
        if ( errno == EINTR )
            return;
        fprintf(stderr, "klogd: Error return from sys_sycall: " \
            "%d - %s\n", errno, strerror(errno));
    }
    else
        LogLine(log_buffer, rdcnt);
    return;
}

static void LogProcLine(void)

{
    auto int rdcnt;

    /*
     * Zero-fill the log buffer. This should cure a multitude of
     * problems with klogd logging the tail end of the message buffer
     * which will contain old messages. Then read the kernel messages
     * from the message pseudo-file into this fresh buffer.
     */
    memset(log_buffer, '\0', sizeof(log_buffer));
    if ( (rdcnt = read(kmsg, log_buffer, sizeof(log_buffer)-1)) < 0 )
    {
        if ( errno == EINTR )
            return;
        Syslog(LOG_ERR, "Cannot read proc file system: %d - %s.", \
            errno, strerror(errno));
    }
    else
        LogLine(log_buffer, rdcnt);
}
```

```

return;
}

int main(argc, argv)

int argc;

char *argv[];

{
    auto intch,
    use_output = 0;

    auto char*log_level = (char *) 0,
    *output = (char *) 0;

    #ifndef TESTING
    pid_t ppid = getpid();
    chdir ("/");
    #endif
    /* Parse the command-line. */
    while ((ch = getopt(argc, argv, "c:df:ilk:nopsvx2")) != EOF)
    switch((char)ch)
    {
        case '2':/* Print lines with symbols twice. */
            symbols_twice = 1;
            break;
        case 'c':/* Set console message level. */
            log_level = optarg;
            break;
        case 'd':/* Activity debug mode. */
            debugging = 1;
            break;
        case 'f':/* Define an output file. */
            output = optarg;
            use_output++;
            break;
        case 'i':/* Reload module symbols. */
            SignalDaemon(SIGUSR1);
            return(0);
        case 'I':
            SignalDaemon(SIGUSR2);
            return(0);
        case 'k':/* Kernel symbol file. */
            symfile = optarg;
            break;
        case 'n':/* don't fork */
            no_fork++;
            break;
        case 'o':/* One-shot mode. */
            one_shot = 1;
            break;
        case 'p':
            SetParanoiaLevel(1);/* Load symbols on oops. */

```

```

break;
    case 's':/* Use syscall interface. */
use_syscall = 1;
break;
    case 'v':
printf("klogd %s.%s\n", VERSION, PATCHLEVEL);
exit (1);
    case 'x':
symbol_lookup = 0;
break;
}

/* Set console logging level. */
if ( log_level != (char *) 0 )
{
if ( (strlen(log_level) > 1) || \
    (strchr("12345678", *log_level) == (char *) 0) )
{
fprintf(stderr, "klogd: Invalid console logging "
"level <%s> specified.\n", log_level);
return(1);
}
console_log_level = *log_level - '0';
}

#ifdef TESTING
/*
 * The following code allows klogd to auto-background itself.
 * What happens is that the program forks and the parent quits.
 * The child closes all its open file descriptors, and issues a
 * call to setsid to establish itself as an independent session
 * immune from control signals.
 *
 * fork() is only called if it should run in daemon mode, fork is
 * not disabled with the command line argument and there's no
 * such process running.
 */
if ( (!one_shot) && (!no_fork) )
{
if (!check_pid(PidFile))
{
signal (SIGTERM, doexit);
if ( fork() == 0 )
{
auto int fl;
int num_fds = getdtablesize();

signal (SIGTERM, SIG_DFL);

/* This is the child closing its file descriptors. */
for (fl= 0; fl <= num_fds; ++fl)
{

```

```

if ( fileno(stdout) == fl && use_output )
if ( strcmp(output, "-") == 0 )
continue;
close(fl);
}

setsid();
}
else
{
/*
* Parent process
*/
sleep(300);
/*
* Not reached unless something major went wrong.
*/
exit(1);
}
}
else
{
fputs("klogd: Already running.\n", stderr);
exit(1);
}
}

/* tuck my process id away */
if (!check_pid(PidFile))
{
if (!write_pid(PidFile))
Terminate();
}
else
{
fputs("klogd: Already running.\n", stderr);
Terminate();
}
}
#endif

/* Signal setups. */
for (ch= 1; ch < NSIG; ++ch)
signal(ch, SIG_IGN);
signal(SIGINT, stop_daemon);
signal(SIGKILL, stop_daemon);
signal(SIGTERM, stop_daemon);
signal(SIGHUP, stop_daemon);
signal(SIGTSTP, stop_logging);
signal(SIGCONT, restart);
signal(SIGUSR1, reload_daemon);
signal(SIGUSR2, reload_daemon);

/* Open outputs. */

```



```
if ( use_output )
{
if ( strcmp(output, "-") == 0 )
output_file = stdout;
else if ( (output_file = fopen(output, "w")) == (FILE *) 0 )
{
fprintf(stderr, "klogd: Cannot open output file " \
"%s - %s\n", output, strerror(errno));
return(1);
}
}
else
openlog("kernel", 0, LOG_KERN);

/* Handle one-shot logging. */
if ( one_shot )
{
if (symbol_lookup) {
InitKsyms(symfile);
InitMsyms();
}
if ( (logsrc = GetKernelLogSrc()) == kernel )
LogKernelLine();
else
LogProcLine();
Terminate();
}

/* Determine where kernel logging information is to come from. */
#ifdef KLOGD_DELAY
sleep(KLOGD_DELAY);
#endif
logsrc = GetKernelLogSrc();
if (symbol_lookup) {
InitKsyms(symfile);
InitMsyms();
}

#ifdef TESTING
if (getpid() != ppid)
kill (ppid, SIGTERM);
#endif

/* The main loop. */
while (1)
{
if ( change_state )
ChangeLogging();
switch ( logsrc )
{
case kernel:
LogKernelLine();
break;
```

```

case proc:
LogProcLine();
break;
    case none:
pause();
break;
}
}
}
/*
* Local variables:
* c-indent-level: 8
* c-basic-offset: 8
* tab-width: 8
* End:
*/
/*
* Copyright (c) 1983, 1988 Regents of the University of California.
* All rights reserved.
*
* Redistribution and use in source and binary forms are permitted
* provided that the above copyright notice and this paragraph are
* duplicated in all such forms and that any documentation,
* advertising materials, and other materials related to such
* distribution and use acknowledge that the software was developed
* by the University of California, Berkeley. The name of the
* University may not be used to endorse or promote products derived
* from this software without specific prior written permission.
* THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR
* IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
* WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
*/

#ifdef lint
char copyright2[] =
"@(#) Copyright (c) 1983, 1988 Regents of the University of California.\n\
All rights reserved.\n";
#endif /* not lint */

#ifdef lint
static char sccsid[] = "@(#)syslogd.c5.27 (Berkeley) 10/10/88";
#endif /* not lint */

/*
* syslogd -- log system messages
*
* This program implements a system log. It takes a series of lines.
* Each line may have a priority, signified as "<n>" as
* the first characters of the line. If this is
* not present, a default priority is used.
*
* To kill syslogd, send a signal 15 (terminate). A signal 1 (hup) will
* cause it to reread its configuration file.
*
* Defined Constants:

```

*
* MAXLINE -- the maximum line length that can be handled.
* DEFUPRI -- the default priority for user messages
* DEFSPRI -- the default priority for kernel messages
*
* Author: Eric Allman
* extensive changes by Ralph Campbell
* more extensive changes by Eric Allman (again)
*
* Steve Lord: Fix UNIX domain socket code, added linux kernel logging
* change defines to
* SYSLOG_INET- listen on a UDP socket
* SYSLOG_UNIXAF- listen on unix domain socket
* SYSLOG_KERNEL- listen to linux kernel
*
* Mon Feb 22 09:55:42 CST 1993: Dr. Wettstein
* Additional modifications to the source. Changed priority scheme
* to increase the level of configurability. In its stock configuration
* syslogd no longer logs all messages of a certain priority and above
* to a log file. The * wildcard is supported to specify all priorities.
* Note that this is a departure from the BSD standard.
*
* Syslogd will now listen to both the inetd and the unixd socket. The
* strategy is to allow all local programs to direct their output to
* syslogd through the unixd socket while the program listens to the
* inetd socket to get messages forwarded from other hosts.
*
* Fri Mar 12 16:55:33 CST 1993: Dr. Wettstein
* Thanks to Stephen Tweedie (dcs.ed.ac.uk!sct) for helpful bug-fixes
* and an enlightened commentary on the prioritization problem.
*
* Changed the priority scheme so that the default behavior mimics the
* standard BSD. In this scenario all messages of a specified priority
* and above are logged.
*
* Add the ability to specify a wildcard (=) as the first character
* of the priority name. Doing this specifies that ONLY messages with
* this level of priority are to be logged. For example:
*
* *.=debug/usr/adm/debug
*
* Would log only messages with a priority of debug to the /usr/adm/debug
* file.
*
* Providing an * as the priority specifies that all messages are to be
* logged. Note that this case is degenerate with specifying a priority
* level of debug. The wildcard * was retained because I believe that
* this is more intuitive.
*
* Thu Jun 24 11:34:13 CDT 1993: Dr. Wettstein
* Modified sources to incorporate changes in libc4.4. Messages from
* syslog are now null-terminated, syslogd code now parses messages
* based on this termination scheme. Linux as of libc4.4 supports the
* fsync system call. Modified code to fsync after all writes to

*log files.
*
* Sat Dec 11 11:59:43 CST 1993: Dr. Wettstein
*Extensive changes to the source code to allow compilation with no
*complaints with -Wall.
*
*Reorganized the facility and priority name arrays so that they
*compatible with the syslog.h source found in /usr/include/syslog.h.
*NOTE that this should really be changed. The reason I do not
*allow the use of the values defined in syslog.h is on account of
*the extensions made to allow the wildcard character in the
*priority field. To fix this properly one should malloc an array,
*copy the contents of the array defined by syslog.h and then
*make whatever modifications that are desired. Next round.
*
* Thu Jan 6 12:07:36 CST 1994: Dr. Wettstein
*Added support for proper decomposition and re-assembly of
*fragment messages on UNIX domain sockets. Lack of this capability
*was causing 'partial' messages to be output. Since facility and
*priority information is encoded as a leader on the messages this
*was causing lines to be placed in erroneous files.
*
*Also added a patch from Shane Alderton (shane@ion.apana.org.au) to
*correct a problem with syslogd dumping core when an attempt was made
*to write log messages to a logged-on user. Thank you.
*
*Many thanks to Juha Virtanen (jiivee@hut.fi) for a series of
*interchanges which lead to the fixing of problems with messages set
*to priorities of none and emerg. Also thanks to Juha for a patch
*to exclude users with a class of LOGIN from receiving messages.
*
*Shane Alderton provided an additional patch to fix zombies which
*were conceived when messages were written to multiple users.
*
* Mon Feb 6 09:57:10 CST 1995: Dr. Wettstein
*Patch to properly reset the single priority message flag. Thanks
*to Christopher Gori for spotting this bug and forwarding a patch.
*
* Wed Feb 22 15:38:31 CST 1995: Dr. Wettstein
*Added version information to startup messages.
*
*Added defines so that paths to important files are taken from
*the definitions in paths.h. Hopefully this will insure that
*everything follows the FSSTND standards. Thanks to Chris Metcalf
*for a set of patches to provide this functionality. Also thanks
*Elias Levy for prompting me to get these into the sources.
*
* Wed Jul 26 18:57:23 MET DST 1995: Martin Schulze
*Linux' gethostname only returns the hostname and not the fqdn as
*expected in the code. But if you call hostname with an fqdn then
*gethostname will return an fqdn, so we have to mention that. This
*has been changed.
*
*The 'LocalDomain' and the hostname of a remote machine is
*converted to lower case, because the original caused some

*inconsistency, because the (at least my) nameserver did respond an
*fqdn containing of upper- _and_ lowercase letters while
*'LocalDomain' consisted only of lowercase letters and that didn't
*match.
*
* Sat Aug 5 18:59:15 MET DST 1995: Martin Schulze
*Now no messages that were received from any remote host are sent
*out to another. At my domain this missing feature caused ugly
*syslog-loops, sometimes.
*
*Remember that no message is sent out. I can't figure out any
*scenario where it might be useful to change this behavior and to
*send out messages to other hosts than the one from which we
*received the message, but I might be shortsighted. :-/
*
* Thu Aug 10 19:01:08 MET DST 1995: Martin Schulze
*Added my pidfile.[ch] to it to perform a better handling with
*pidfiles. Now both, syslogd and klogd, can only be started
*once. They check the pidfile.
*
* Sun Aug 13 19:01:41 MET DST 1995: Martin Schulze
*Add an addition to syslog.conf's interpretation. If a priority
*begins with an exclamation mark (!) the normal interpretation
of the priority is inverted: "." is the same as ".none", ".*=info"
*don't logs the info priority, ".*crit" won't log any message with
*the priority crit or higher. For example:
*
mail.;mail.!=info/usr/adm/mail
*
*Would log all messages of the facility mail except those with
*the priority info to /usr/adm/mail. This makes the syslogd
*much more flexible.
*
*Defined TABLE_ALLPRI=255 and changed some occurrences.
*
* Sat Aug 19 21:40:13 MET DST 1995: Martin Schulze
*Making the table of facilities and priorities while in debug
*mode more readable.
*
*If debugging is turned on, printing the whole table of
*facilities and priorities every hexadecimal or 'X' entry is
*now 2 characters wide.
*
*The number of the entry is prepended to each line of
*facilities and priorities, and F_UNUSED lines are not shown
*anymore.
*
*Corrected some #ifdef SYSV's.
*
* Mon Aug 21 22:10:35 MET DST 1995: Martin Schulze
*Corrected a strange behavior during parsing of configuration
*file. The original BSD syslogd doesn't understand spaces as
*separators between specifier and action. This syslogd now
*understands them. The old behavior caused some confusion over

*the Linux community.
*
* Thu Oct 19 00:02:07 MET 1995: Martin Schulze
*The default behavior has changed for security reasons. The
*syslogd will not receive any remote message unless you turn
*reception on with the "-r" option.
*
*Not defining SYSLOG_INET will result in not doing any network
*activity, i.e. not sending or receiving messages. I changed
*this because the old idea is implemented with the "-r" option
*and the old thing didn't work anyway.
*
* Thu Oct 26 13:14:06 MET 1995: Martin Schulze
*Added another logfile type F_FORW_UNKN. The problem I ran into
*was a name server that runs on my machine and a forwarder of
*kern.crit to another host. The hosts address can only be
*fetched using the nameserver. But named is started after
*syslogd, so syslogd complained.
*
*This logfile type will retry to get the address of the
*hostname ten times and then complain. This should be enough to
*get the named up and running during boot sequence.
*
* Fri Oct 27 14:08:15 1995: Dr. Wettstein
*Changed static array of logfiles to a dynamic array. This
*can grow during process.
*
* Fri Nov 10 23:08:18 1995: Martin Schulze
*Inserted a new tabular sys_h_errlist that contains plain text
*for error codes that are returned from the net subsystem and
*stored in h_errno. I have also changed some wrong lookups to
*sys_errlist.
*
* Wed Nov 22 22:32:55 1995: Martin Schulze
*Added the fabulous strip-domain feature that allows us to
*strip off (several) domain names from the fqdn and only log
*the simple hostname. This is useful if you're in a LAN that
*has a central log server and also different domains.
*
*I have also added the -l switch to define hosts as
*local. These will get logged with their simple hostname, too.
*
* Thu Nov 23 19:02:56 MET DST 1995: Martin Schulze
*Added the possibility to omit fsyncing of logfiles after every
*write. This will give some performance back if you have
*programs that log in a very verbose manner (like innd or
*smartlist). Thanks to Stephen R. van den Berg <srb@cuci.nl>
*for the idea.
*
* Thu Jan 18 11:14:36 CST 1996: Dr. Wettstein
*Added patches from beta-testers to stop compile error. Also
*added removal of pid file as part of termination cleanup.
*
* Wed Feb 14 12:42:09 CST 1996: Dr. Wettstein
*Allowed forwarding of messages received from remote hosts to

*be controlled by a command-line switch. Specifying -h allows
*forwarding. The default behavior is to disable forwarding of
*messages which were received from a remote host.
*
*Parent process of syslogd does not exit until child process has
finished initialization process. This allows rc. startup to
*pause until syslogd facility is up and operating.
*
*Re-arranged the select code to move UNIX domain socket accepts
*to be processed later. This was a contributed change which
*has been proposed to correct the delays sometimes encountered
*when syslogd starts up.
*
*Minor code cleanups.
*
* Thu May 2 15:15:33 CDT 1996: Dr. Wettstein
*Fixed bug in init function which resulted in file descriptors
*being orphaned when syslogd process was re-initialized with SIGHUP
*signal. Thanks to Edvard Tuinder
*(Edvard.Tuinder@praseodymium.cistron.nl) for putting me on the
*trail of this bug. I am amazed that we didn't catch this one
*before now.
*
* Tue May 14 00:03:35 MET DST 1996: Martin Schulze
*Corrected a mistake that causes the syslogd to stop logging at
*some virtual consoles under Linux. This was caused by checking
*the wrong error code. Thanks to Michael Nonweiler
*<mrn20@hermes.cam.ac.uk> for sending me a patch.
*
* Mon May 20 13:29:32 MET DST 1996: Miquel van Smoorenburg <miquels@cistron.nl>
*Added continuation line supported and fixed a bug in
*the init() code.
*
* Tue May 28 00:58:45 MET DST 1996: Martin Schulze
*Corrected behaviour of blocking pipes - i.e. the whole system
*hung. Michael Nonweiler <mrn20@hermes.cam.ac.uk> has sent us
*a patch to correct this. A new logfile type F_PIPE has been
*introduced.
*
* Mon Feb 3 10:12:15 MET DST 1997: Martin Schulze
*Corrected behaviour of logfiles if the file can't be opened.
*There was a bug that causes syslogd to try to log into non
*existing files which ate cpu power.
*
* Sun Feb 9 03:22:12 MET DST 1997: Martin Schulze
*Modified syslogd.c to not kill itself which confuses bash 2.0.
*
* Mon Feb 10 00:09:11 MET DST 1997: Martin Schulze
*Improved debug code to decode the numeric facility/priority
*pair into textual information.
*
* Tue Jun 10 12:35:10 MET DST 1997: Martin Schulze
*Corrected freeing of logfiles. Thanks to Jos Vos <jos@xos.nl>
*for reporting the bug and sending an idea to fix the problem.

*
* Tue Jun 10 12:51:41 MET DST 1997: Martin Schulze
*Removed sleep(10) from parent process. This has caused a slow
*startup in former times - and I don't see any reason for this.
*
* Sun Jun 15 16:23:29 MET DST 1997: Michael Alan Dorman
*Some more glibc patches made by <mdorman@debian.org>.
*
* Thu Jan 1 16:04:52 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Applied patch from Herbert Thielen <Herbert.Thielen@lpr.e-technik.tu-muenchen.de>.
*This included some balance parentheses for emacs and a bug in
*the exclamation mark handling.
*
*Fixed small bug which caused syslogd to write messages to the
*wrong logfile under some very rare conditions. Thanks to
*Herbert Xu <herbert@gondor.apana.org.au> for fiddling this out.
*
* Thu Jan 8 22:46:35 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Reworked one line of the above patch as it prevented syslogd
*from binding the socket with the result that no messages were
*forwarded to other hosts.
*
* Sat Jan 10 01:33:06 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Fixed small bugs in F_FORW_UNKN mechanism. Thanks to Torsten
*Neumann <torsten@londo.rhein-main.de> for pointing me to it.
*
* Mon Jan 12 19:50:58 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Modified debug output concerning remote reception.
*
* Mon Feb 23 23:32:35 CET 1998: Topi Miettinen <Topi.Miettinen@ml.tele.fi>
*Re-worked handling of Unix and UDP sockets to support closing /
*opening of them in order to have it open only if it is needed
*either for forwarding to a remote host or by reception from
*the network.
*
* Wed Feb 25 10:54:09 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Fixed little comparison mistake that prevented the MARK
*feature to work properly.
*
* Wed Feb 25 13:21:44 CET 1998: Martin Schulze <joey@infodrom.north.de>
*Corrected Topi's patch as it prevented forwarding during
*startup due to an unknown LogPort.
*
* Sat Oct 10 20:01:48 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Added support for TESTING define which will turn syslogd into
*stdio-mode used for debugging.
*
* Sun Oct 11 20:16:59 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Reworked the initialization/fork code. Now the parent
*process activates a signal handler which the daughter process
*will raise if it is initialized. Only after that one the
*parent process may exit. Otherwise klogd might try to flush
*its log cache while syslogd can't receive the messages yet.
*
* Mon Oct 12 13:30:35 CEST 1998: Martin Schulze <joey@infodrom.north.de>

*Redirected some error output with regard to argument parsing to
*stderr.
*
* Mon Oct 12 14:02:51 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Applied patch provided vom Topi Miettinen with regard to the
*people from OpenBSD. This provides the additional '-a'
*argument used for specifying additional UNIX domain sockets to
*listen to. This is been used with chroot()'ed named's for
*example. See for <http://www.psionic.com/papers/dns.html>
*
* Mon Oct 12 18:29:44 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Added 'ftp' facility which was introduced in glibc version 2.
*It's #ifdef'ed so won't harm with older libraries.
*
* Mon Oct 12 19:59:21 MET DST 1998: Martin Schulze <joey@infodrom.north.de>
*Code cleanups with regard to bsd -> posix transition and
*stronger security (buffer length checking). Thanks to Topi
*Miettinen <tom@medialab.sonera.net>
*. index() --> strchr()
*. sprintf() --> snprintf()
*. bcopy() --> memcpy()
*. bzero() --> memset()
*. UNAMESZ --> UT_NAMESIZE
*. sys_errlist --> strerror()
*
* Mon Oct 12 20:22:59 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Added support for setutent()/getutent()/endutend() instead of
*binary reading the UTMP file. This is the the most portable
*way. This allows /var/run/utmp format to change, even to a
*real database or utmp daemon. Also if utmp file locking is
*implemented in libc, syslog will use it immediately. Thanks
*to Topi Miettinen <tom@medialab.sonera.net>.
*
* Mon Oct 12 20:49:18 MET DST 1998: Martin Schulze <joey@infodrom.north.de>
*Avoid logging of SIGCHLD when syslogd is in the process of
*exiting and closing its files. Again thanks to Topi.
*
* Mon Oct 12 22:18:34 CEST 1998: Martin Schulze <joey@infodrom.north.de>
*Modified printline() to support 8bit characters - such as
*russian letters. Thanks to Vldas Lapinskas <lapinskas@mail.iae.lt>.
*
* Sat Nov 14 02:29:37 CET 1998: Martin Schulze <joey@infodrom.north.de>
*``-m 0" now turns of MARK logging entirely.
*
* Tue Jan 19 01:04:18 MET 1999: Martin Schulze <joey@infodrom.north.de>
*Finally fixed an error with '-a' processing, thanks to Topi
*Miettinen <tom@medialab.sonera.net>.
*
* Sun May 23 10:08:53 CEST 1999: Martin Schulze <joey@infodrom.north.de>
*Removed superflous call to utmpname(). The path to the utmp
*file is defined in the used libc and should not be hardcoded
*into the syslogd binary referring the system it was compiled on.
*
* Sun Sep 17 21:26:16 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>

*Don't close open sockets upon reload. Thanks to Bill Nottingham.
*

* Mon Sep 18 09:10:47 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>
*Fixed bug in printchopped() that caused syslogd to emit *kern.emerg messages when splitting long lines. Thanks to *Daniel Jacobowitz <dan@debian.org> for the fix.
*

* Mon Sep 18 15:33:26 CEST 2000: Martin Schulze <joey@infodrom.ffis.de>
*Removed unixm/unix domain sockets and switch to Datagram Unix *Sockets. This should remove one possibility to play DoS with *syslogd. Thanks to Olaf Kirch <okir@caldera.de> for the patch.
*

* Sun Mar 11 20:23:44 CET 2001: Martin Schulze <joey@infodrom.ffis.de>
*Don't return a closed fd if '-a' is called with a wrong path.
*Thanks to Bill Nottingham <notting@redhat.com> for providing *a patch.
* Thu Apr 13 05:08:10 CEST 2001: Jon Burgess <Jon_Burgess@eur.3com.com>
*Moved the installation of the signal handler up a little bit *so it guaranteed to be available when the child is forked, *hence, fixing a race condition. This used to create problems *with UML and fast machines.
*

* Sat Apr 17 18:03:05 CEST 2004: Steve Grubb <linux_4ever@yahoo.com>
*Correct memory allocation for for commandline arguments in *crunch_list().
*

* Thu Apr 29 12:38:39 CEST 2004: Solar Designer <solar@openwall.com>
*Applied Openwall paranoia patches to improve crunch_list().
*

* Tue May 4 16:47:30 CEST 2004: Solar Designer <solar@openwall.com>
*Ensure that "len" is not placed in a register, and that the *endtty() signal handler is not installed too early which could *cause a segmentation fault or worse.
*

* Tue May 4 16:52:01 CEST 2004: Solar Designer <solar@openwall.com>
*Adjust the size of a variable to prevent a buffer overflow *should _PATH_DEV ever contain something different than "/dev".
*

* Tue Nov 2 20:28:23 CET 2004: Colin Phipps <cph@cph.demon.co.uk>
*Don't block on the network socket, in case a packet gets lost *between select and recv.
*

* Sun Nov 7 12:28:47 CET 2004: Martin Schulze <joey@infodrom.org>
*Discard any timestamp information found in received syslog *messages. This will affect local messages sent from a *different timezone.
*

* Sun Nov 7 13:47:00 CET 2004: Martin Schulze <joey@infodrom.org>
*Remove trailing newline when forwarding messages.
*

* Thu May 25 09:47:38 CEST 2006: Martin Schulze <joey@infodrom.org>
*Reset the 'restart' flag immediately after entering the *restart code, so that subsequent SIGHUPs are able to set it *again and cause a new restart. This fixes a race condition

```

*when somebody sends tons of HUP signals.
*
* Thu May 24 15:24:49 CEST 2007: Martin Schulze <joey@infodrom.org>
*Ignore errors caused by filled up disks so that the log
*continues to be written as soon as space becomes available
*again.
*
* Sat May 26 10:05:05 CEST 2007: Martin Schulze <joey@infodrom.org>
*Only try to gather the local domain name when messages are to
*be received from the network, it's not needed otherwise.
*
* Sat May 26 12:22:44 CEST 2007: Martin Schulze <joey@infodrom.org>
*Properly accompany the MARK message with the facility.
*
* Mon May 28 19:44:39 CEST 2007: Martin Schulze <joey@infodrom.org>
*Notify the waiting parent process if the client dies to it
*doesn't wait the entire five minutes.
*
* Wed Jul  4 21:02:22 CEST 2007: Martin Schulze <joey@infodrom.org>
*Open a pipe with O_NOCTTY to avoid them becoming the controlling
*tty and normal files with O_NONBLOCK to avoid blocking.
*
* Fri Sep 10 08:29:04 CEST 2010: Martin Schulze <joey@infodrom.org>
*  Replace strepy with memmove to fix continuation line problems
*  on 64bit architectures, patch by David Couture.
*/

```

```

#define MAXLINE 1024 /* maximum line length */
#define MAXSVLINE 240 /* maximum saved line length */
#define DEFUPRI(LOG_USER|LOG_NOTICE)
#define DEFSPRI(LOG_KERN|LOG_CRIT)
#define TIMERINTVL 30 /* interval for checking flush, mark */

```

```

#define CONT_LINE 1 /* Allow continuation lines */

```

```

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#ifdef SYSV
#include <sys/types.h>
#endif
#include <utmp.h>
#include <ctype.h>
#include <string.h>
#include <setjmp.h>
#include <stdarg.h>
#include <time.h>

```

```

#define SYSLOG_NAMES
#include <sys/syslog.h>
#include <sys/param.h>
#include <sys/errno.h>
#include <sys/ioctl.h>

```

```
#include <sys/stat.h>
#include <sys/wait.h>
#include <sys/socket.h>
#include <sys/file.h>
#ifdef SYSV
#include <fcntl.h>
#else
#include <sys/msgbuf.h>
#endif
#include <sys/uio.h>
#include <sys/un.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <signal.h>

#include <netinet/in.h>
#include <netdb.h>
#include <syscall.h>
#include <arpa/nameser.h>
#include <arpa/inet.h>
#include <resolv.h>
#ifdef TESTING
#include "pidfile.h"
#endif
#include "version.h"

#ifdef __linux__
#include <paths.h>
#endif

#ifdef UTMP_FILE
#ifdef UTMP_FILENAME
#define UTMP_FILE UTMP_FILENAME
#else
#ifdef _PATH_UTMP
#define UTMP_FILE _PATH_UTMP
#else
#define UTMP_FILE "/etc/utmp"
#endif
#endif
#endif

#ifdef _PATH_LOGCONF
#define _PATH_LOGCONF "/etc/syslog.conf"
#endif

#ifdef SYSLOGD_PIDNAME
#ifdef _PATH_LOGPID
#ifdef FSSTND
#define _PATH_LOGPID _PATH_VARRUN SYSLOGD_PIDNAME
#else
#define _PATH_LOGPID "/etc/" SYSLOGD_PIDNAME
#endif
#else
#define _PATH_LOGPID
#endif
#endif
```

```
#if defined(FSSTND)
#define _PATH_LOGPID _PATH_VARRUN "syslogd.pid"
#else
#define _PATH_LOGPID "/etc/syslogd.pid"
#endif
#endif
#endif

#ifndef _PATH_DEV
#define _PATH_DEV "/dev/"
#endif

#ifndef _PATH_CONSOLE
#define _PATH_CONSOLE "/dev/console"
#endif

#ifndef _PATH_TTY
#define _PATH_TTY "/dev/tty"
#endif

#ifndef _PATH_LOG
#define _PATH_LOG "/dev/log"
#endif

/* systemd support */
#define SD_LISTEN_FDS_START 3

char*ConfFile = _PATH_LOGCONF;
char*PidFile = _PATH_LOGPID;
charctty[] = _PATH_CONSOLE;

char**parts;

int inetm = 0;
static int debugging_on = 0;
static int nlogs = -1;
static int restart = 0;
static int socket_from_systemd = 0;

#define MAXFUNIX20

int nfunix = 1;
char *funixn[MAXFUNIX] = { _PATH_LOG };
int funix[MAXFUNIX] = { -1, };

#ifndef UT_NAMESIZE
#define UNAMESZUT_NAMESIZE/* length of a login name */
#else
#define UNAMESZ8/* length of a login name */
#endif
#define MAXUNAMES20/* maximum number of user names */
#define MAXFNAME200/* max file pathname length */

#define INTERNAL_NOPRIOx10/* the "no priority" priority */
```

```

#define TABLE_NOPRI0/* Value to indicate no priority in f_pmask */
#define TABLE_ALLPRI 0xFF /* Value to indicate all priorities in f_pmask */
#define LOG_MARKLOG_MAKEPRI(LOG_NFACILITIES, 0)/* mark "facility" */

#define MAX_PRI191/* Maximum Priority per RFC 3164 */

/*
 * Flags to logmsg().
 */

#define IGN_CONS0x001/* don't print on console */
#define SYNC_FILE0x002/* do fsync on file after printing */
#define ADDDATE0x004/* add a date to the message */
#define MARK0x008/* this message is a mark */

/*
 * This table contains plain text for h_errno errors used by the
 * net subsystem.
 */
const char *sys_h_errlist[] = {
    "No problem",/* NETDB_SUCCESS */
    "Authoritative answer: host not found",/* HOST_NOT_FOUND */
    "Non-authoritative answer: host not found, or serverfail",/* TRY_AGAIN */
    "Non recoverable errors",/* NO_RECOVERY */
    "Valid name, no data record of requested type",/* NO_DATA */
    "no address, look for MX record",/* NO_ADDRESS */
};

/*
 * This structure represents the files that will have log
 * copies printed.
 */

struct filed {
#ifdef SYSV
    structfile *f_next;/* next in linked list */
#endif
    shortf_type;/* entry type, see below */
    shortf_file;/* file descriptor */
    time_tf_time;/* time this was last written */
    u_charf_pmask[LOG_NFACILITIES+1];/* priority mask */
    union {
        charf_undef[UNAMESZ+1];
        struct {
            charf_hname[MAXHOSTNAMELEN+1];
            struct sockaddr_in f_addr;
        } f_forw;/* forwarding address */
        charf_fname[MAXFNAME];
    } f_un;
    charf_prevline[MAXSVLINE];/* last message logged */
    charf_lasttime[16];/* time of last occurrence */
    charf_prevhost[MAXHOSTNAMELEN+1];/* host from which recd. */
    intf_prevpri;/* pri of f_prevline */
    intf_prevlen;/* length of f_prevline */
    intf_prevcount;/* repetition cnt of prevline */

```

```

intf_repeatcount;/* number of "repeated" msgs */
intf_flags;/* store some additional flags */
};

/*
 * Intervals at which we flush out "message repeated" messages,
 * in seconds after previous message is logged. After each flush,
 * we move to the next interval until we reach the largest.
 */
intrepeatinterval[] = { 30, 60 };/* # of secs before flush */
#defineMAXREPEAT ((sizeof(repeatinterval) / sizeof(repeatinterval[0])) - 1)
#defineREPEATTIME(f)((f)->f_time + repeatinterval[(f)->f_repeatcount])
#defineBACKOFF(f){ if (++(f)->f_repeatcount > MAXREPEAT) \
    (f)->f_repeatcount = MAXREPEAT; \
}
#ifdef SYSLOG_INET
#define INET_SUSPEND_TIME 180/* equal to 3 minutes */
#define INET_RETRY_MAX 10/* maximum of retries for gethostbyname() */
#endif

#define LIST_DELIMITER':'/* delimiter between two hosts */

/* values for f_type */
#define F_UNUSED0/* unused entry */
#define F_FILE1/* regular file */
#define F_TTY2/* terminal */
#define F_CONSOLE3/* console terminal */
#define F_FORW4/* remote machine */
#define F_USERS5/* list of users */
#define F_WALL6/* everyone logged on */
#define F_FORW_SUSP7/* suspended host forwarding */
#define F_FORW_UNKN8/* unknown host forwarding */
#define F_PIPE9/* named pipe */
char*TypeNames[] = {
    "UNUSED","FILE","TTY","CONSOLE",
    "FORW","USERS","WALL","FORW(SUSPENDED)",
    "FORW(UNKNOWN)","PIPE"
};

structfiled *Files = (struct filed *) 0;
structfiled consfile;

struct code {
    char*c_name;
    intc_val;
};

struct codePriNames[] = {
    {"alert",LOG_ALERT},
    {"crit",LOG_CRIT},
    {"debug",LOG_DEBUG},
    {"emerg",LOG_EMERG},
    {"err",LOG_ERR},
    {"error",LOG_ERR}/* DEPRECATED */

```

```

{"info",LOG_INFO},
{"none",INTERNAL_NOPRI},/* INTERNAL */
{"notice",LOG_NOTICE},
{"panic",LOG_EMERG},/* DEPRECATED */
{"warn",LOG_WARNING},/* DEPRECATED */
{"warning",LOG_WARNING},
{"*",TABLE_ALLPRI},
{NULL,-1}
};

struct codeFacNames[] = {
{"auth",    LOG_AUTH},
{"authpriv", LOG_AUTHPRIV},
{"cron",    LOG_CRON},
{"daemon",  LOG_DAEMON},
{"kern",    LOG_KERN},
{"lpr",     LOG_LPR},
{"mail",    LOG_MAIL},
{"mark",    LOG_MARK},/* INTERNAL */
{"news",    LOG_NEWS},
{"security", LOG_AUTH},/* DEPRECATED */
{"syslog",  LOG_SYSLOG},
{"user",    LOG_USER},
{"uucp",    LOG_UUCP},
#ifdef LOG_FTP
{"ftp",     LOG_FTP},
#endif
{"local0",  LOG_LOCAL0},
{"local1",  LOG_LOCAL1},
{"local2",  LOG_LOCAL2},
{"local3",  LOG_LOCAL3},
{"local4",  LOG_LOCAL4},
{"local5",  LOG_LOCAL5},
{"local6",  LOG_LOCAL6},
{"local7",  LOG_LOCAL7},
{NULL,     -1},
};

intDebug;/* debug flag */
charLocalHostName[MAXHOSTNAMELEN+1];/* our hostname */
char*LocalDomain;/* our local domain name */
char*emptystring = "";
intInetInuse = 0;/* non-zero if INET sockets are being used */
intfinet = -1;/* Internet datagram socket */
intLogPort;/* port number for INET connections */
intInitialized = 0;/* set when we have initialized ourselves */
intMarkInterval = 20 * 60;/* interval between marks in seconds */
intMarkSeq = 0;/* mark sequence number */
intNoFork = 0;/* don't fork - don't run in daemon mode */
intAcceptRemote = 0;/* receive messages that come via UDP */
char**StripDomains = NULL;/* these domains may be stripped before writing logs */
char**LocalHosts = NULL;/* these hosts are logged with their hostname */
intNoHops = 1;/* Can we bounce syslog messages through an
    intermediate host. */

```



```

externint errno;

/* Function prototypes. */
int main(int argc, char **argv);
char **crunch_list(char *list);
int usage(void);
void unttty(void);
void printchopped(const char *hname, char *msg, int len, int fd);
void printline(const char *hname, char *msg);
void printsys(char *msg);
void logmsg(int pri, char *msg, const char *from, int flags);
void fprintlog(register struct filed *f, char *from, int flags, char *msg);
void endtty();
void wallmsg(register struct filed *f, struct iovec *iov);
void reapchild();
const char *cvthname(struct sockaddr_in *f);
void domark();
void debug_switch();
void logerror(char *type);
void die(int sig);
#ifdef TESTING
void doexit(int sig);
#endif
void init();
void cfln(char *line, register struct filed *f);
int decode(char *name, struct code *codetab);
#define dprintf mydprintf
static void dprintf(char *, ...);
static void allocate_log(void);
void sighup_handler();
/* systemd support */
int sd_listen_fds(void);

#ifdef SYSLOG_UNIXAF
static int create_unix_socket(const char *path);
#endif
#ifdef SYSLOG_INET
static int create_inet_socket();
#endif

int main(argc, argv)
int argc;
char **argv;
{
    register int i;
    register char *p;
#ifdef TESTING
    ssize_t msglen;
#endif
#ifdef !defined(__GLIBC__)
    int len, num_fds;
#else /* __GLIBC__ */
#ifdef TESTING
    socklen_t len;

```

```

#endif
int num_fds;
#endif /* __GLIBC__ */
/*
 * It took me quite some time to figure out how this is
 * supposed to work so I guess I should better write it down.
 * unixm is a list of file descriptors from which one can
 * read(). This is in contrary to readfds which is a list of
 * file descriptors where activity is monitored by select()
 * and from which one cannot read(). -Joey
 *
 * Changed: unixm is gone, since we now use datagram unix sockets.
 * Hence we recv() from unix sockets directly (rather than
 * first accept()ing connections on them), so there's no need
 * for separate book-keeping. --okir
 */
fd_set readfds;

#ifdef TESTING
intfd;
#ifdef SYSLOG_INET
struct sockaddr_in frominet;
char *from;
#endif
pid_t ppid = getpid();
#endif
int ch;
struct hostent *hent;

char line[MAXLINE + 1];
extern int optind;
extern char *optarg;
int maxfds;

#ifdef TESTING
chdir ("/");
#endif
for (i = 1; i < MAXFUNIX; i++) {
    funixn[i] = "";
    funix[i] = -1;
}

while ((ch = getopt(argc, argv, "a:dhf:l:m:np:rs:v")) != EOF)
    switch((char)ch) {
        case 'a':
            if (nfunix < MAXFUNIX)
                funixn[nfunix++] = optarg;
            else
                fprintf(stderr, "Out of descriptors, ignoring %s\n", optarg);
            break;
        case 'd': /* debug */
            Debug = 1;
            break;
        case 'f': /* configuration file */
            ConfFile = optarg;

```

```
break;
case 'h':
NoHops = 0;
break;
case 'l':
if (LocalHosts) {
fprintf(stderr, "Only one -l argument allowed," \
"the first one is taken.\n");
break;
}
LocalHosts = crunch_list(optarg);
break;
case 'm':/* mark interval */
MarkInterval = atoi(optarg) * 60;
break;
case 'n':/* don't fork */
NoFork = 1;
break;
case 'p':/* path to regular log socket */
funixn[0] = optarg;
break;
case 'r':/* accept remote messages */
AcceptRemote = 1;
break;
case 's':
if (StripDomains) {
fprintf(stderr, "Only one -s argument allowed," \
"the first one is taken.\n");
break;
}
StripDomains = crunch_list(optarg);
break;
case 'v':
printf("syslogd %s.%s\n", VERSION, PATCHLEVEL);
exit (0);
case '?':
default:
usage();
}
if ((argc == optind))
usage();

#ifdef TESTING
if ( !(Debug || NoFork) )
{
dprintf("Checking pidfile.\n");
if (!check_pid(PidFile))
{
signal (SIGTERM, doexit);
if (fork()) {
/*
* Parent process
*/
sleep(300);
```

```

/*
 * Not reached unless something major went wrong. 5
 * minutes should be a fair amount of time to wait.
 * Please note that this procedure is important since
 * the father must not exit before syslogd isn't
 * initialized or the klogd won't be able to flush its
 * logs. -Joey
 */
exit(1);
}

/* We keep stdout and stderr open in case we have to emit something */
close(0);
i = 3;

/* Trying to pass on LISTEN_PID with appropriate value */
const char *e;
unsigned long l;
char buf[24] = { '\0' };
int sd_fds;

if ( (e = getenv("LISTEN_PID")) ) {
    errno = 0;
    l = strtoul(e, NULL, 10);
    if (errno == 0) {
        if (getppid() == (pid_t)l) {
            snprintf(buf, sizeof(buf), "%d", (int)getpid());
            setenv("LISTEN_PID", buf, 1);
        }
    }
}

signal(SIGTERM, SIG_DFL);
num_fds = getdtablesize();

/* close all further fds except of the fds provided by systemd */
sd_fds = sd_listen_fds();
if (sd_fds > 0) {
    i = SD_LISTEN_FDS_START + sd_fds;
}
for ( ; i < num_fds; i++)
    (void) close(i);
unTTY();
}
else
{
    fputs("syslogd: Already running.\n", stderr);
    exit(1);
}
}
else
#endif
debugging_on = 1;
#ifdef SYSV
else

```

```

setlinebuf(stdout);
#endif

#ifdef TESTING
/* tuck my process id away */
if ( !Debug )
{
dprintf("Writing pidfile.\n");
if (!check_pid(PidFile))
{
if (!write_pid(PidFile))
{
dprintf("Can't write pid.\n");
if (getpid() != ppid)
kill (ppid, SIGTERM);
exit(1);
}
}
else
{
dprintf("Pidfile (and pid) already exist.\n");
if (getpid() != ppid)
kill (ppid, SIGTERM);
exit(1);
}
} /* if ( !Debug ) */
#endif

consfile.f_type = F_CONSOLE;
(void) strcpy(consfile.f_un.f_fname, cty);
(void) gethostname(LocalHostName, sizeof(LocalHostName));
LocalDomain = emptystring;
if ( (p = strchr(LocalHostName, '.')) ) {
*p++ = '\0';
LocalDomain = p;
}
else if ( AcceptRemote )
{
/*
 * It's not clearly defined whether gethostname()
 * should return the simple hostname or the fqdn. A
 * good piece of software should be aware of both and
 * we want to distribute good software.  Joey
 *
 * Good software also always checks its return values...
 * If syslogd starts up before DNS is up & /etc/hosts
 * doesn't have LocalHostName listed, gethostbyname will
 * return NULL.
 */
hent = gethostbyname(LocalHostName);
if ( hent )
snprintf(LocalHostName, sizeof(LocalHostName), "%s", hent->h_name);

if ( (p = strchr(LocalHostName, '.')) )

```

```

{
    *p++ = '\0';
    LocalDomain = p;
}
}

/*
 * Convert to lower case to recognize the correct domain laterly
 */
for (p = (char *)LocalDomain; *p ; p++)
    if (isupper(*p))
        *p = tolower(*p);

(void) signal(SIGTERM, die);
(void) signal(SIGINT, Debug ? die : SIG_IGN);
(void) signal(SIGQUIT, Debug ? die : SIG_IGN);
(void) signal(SIGCHLD, reapchild);
(void) signal(SIGALRM, domark);
(void) signal(SIGUSR1, Debug ? debug_switch : SIG_IGN);
(void) signal(SIGXFSZ, SIG_IGN);
(void) alarm(TIMERINTVL);

/* Create a partial message table for all file descriptors. */
num_fds = getdtablesize();
dprintf("Allocated parts table for %d file descriptors.\n", num_fds);
if ( ( parts = (char **) malloc(num_fds * sizeof(char *)) ) == \
    (char **) 0 )
{
    logerror("Cannot allocate memory for message parts table.");
#ifdef TESTING
    if (getpid() != ppid)
        kill (ppid, SIGTERM);
#endif
    die(0);
}
for(i= 0; i < num_fds; ++i)
    parts[i] = (char *) 0;

dprintf("Starting.\n");
init();
#ifdef TESTING
if ( Debug )
{
    dprintf("Debugging disabled, SIGUSR1 to turn on debugging.\n");
    debugging_on = 0;
}
}
/*
 * Send a signal to the parent to it can terminate.
 */
if (getpid() != ppid)
    kill (ppid, SIGTERM);
#endif

/* Main loop begins here. */
for (;;) {

```

```

int nfds;
errno = 0;
FD_ZERO(&readfds);
maxfds = 0;
#ifdef SYSLOG_UNIXAF
#ifndef TESTING
/*
 * Add the Unix Domain Sockets to the list of read
 * descriptors.
 */
/* Copy master connections */
for (i = 0; i < nfunix; i++) {
    if (funix[i] != -1) {
        FD_SET(funix[i], &readfds);
        if (funix[i] > maxfds) maxfds = funix[i];
    }
}
#endif
#endif
#ifdef SYSLOG_INET
#ifndef TESTING
/*
 * Add the Internet Domain Socket to the list of read
 * descriptors.
 */
if ( InetInuse && AcceptRemote ) {
    FD_SET(inetm, &readfds);
    if (inetm > maxfds) maxfds = inetm;
    dprintf("Listening on syslog UDP port.\n");
}
#endif
#endif
#ifdef TESTING
    FD_SET(fileno(stdin), &readfds);
    if (fileno(stdin) > maxfds) maxfds = fileno(stdin);

    dprintf("Listening on stdin. Press Ctrl-C to interrupt.\n");
#endif

if ( debugging_on )
{
    dprintf("Calling select, active file descriptors (max %d): ", maxfds);
    for (nfds = 0; nfds <= maxfds; ++nfds)
        if ( FD_ISSET(nfds, &readfds) )
            dprintf("%d ", nfds);
    dprintf("\n");
}
nfds = select(maxfds+1, (fd_set *) &readfds, (fd_set *) NULL,
    (fd_set *) NULL, (struct timeval *) NULL);
if ( restart )
{
    restart = 0;
    dprintf("\nReceived SIGHUP, reloading syslogd.\n");
    init();
}

```

```

        continue;
    }
    if (nfds == 0) {
        dprintf("No select activity.\n");
        continue;
    }
    if (nfds < 0) {
        if (errno != EINTR)
            logerror("select");
        dprintf("Select interrupted.\n");
        continue;
    }

    if ( debugging_on )
    {
        dprintf("\nSuccessful select, descriptor count = %d, " \
            "Activity on: ", nfds);
        for (nfd=0; nfd <= maxfds; ++nfd)
            if ( FD_ISSET(nfd, &readfds) )
                dprintf("%d ", nfd);
        dprintf("\n");
    }

#ifdef TESTING
#ifdef SYSLOG_UNIXAF
    for (i = 0; i < nfunix; i++) {
        if ((fd = funix[i]) != -1 && FD_ISSET(fd, &readfds)) {
            memset(line, 0, sizeof(line));
            msglen = recv(fd, line, MAXLINE - 2, 0);
            dprintf("Message from UNIX socket: #%d\n", fd);
            if (msglen > 0)
                printchopped(HostName, line, msglen + 2, fd);
            else if (msglen < 0 && errno != EINTR) {
                dprintf("UNIX socket error: %d = %s.\n", \
                    errno, strerror(errno));
                logerror("recvfrom UNIX");
            }
        }
    }
#endif
#endif

#ifdef SYSLOG_INET
    if (InetInuse && AcceptRemote && FD_ISSET(inetm, &readfds)) {
        len = sizeof(frominet);
        memset(line, 0, sizeof(line));
        msglen = recvfrom(finet, line, MAXLINE - 2, 0, \
            (struct sockaddr *) &frominet, &len);
        dprintf("Message from inetd socket: #%d, host: %s\n",
            inetm, inet_ntoa(frominet.sin_addr));
        if (msglen > 0) {
            from = (char *)cvthname(&frominet);
            /*
             * Here we could check if the host is permitted
             * to send us syslog messages. We just have to
             * catch the result of cvthname, look for a dot

```



```

    * and if that doesn't exist, replace the first
    * '\0' with '.' and we have the fqdn in lowercase
    * letters so we could match them against whatever.
    * -Joey
    */
    printchopped(from, line, \
        msglen + 2, finet);
} else if (msglen < 0 && errno != EINTR && errno != EAGAIN) {
    dprintf("INET socket error: %d = %s.\n", \
        errno, strerror(errno));
    logerror("recvfrom inet");
    /* should be harmless now that we set
     * BSDCOMPAT on the socket */
    sleep(1);
}
}
#endif
#else
if ( FD_ISSET(fileno(stdin), &readfds) ) {
    dprintf("Message from stdin.\n");
    memset(line, '\0', sizeof(line));
    line[0] = '.';
    parts[fileno(stdin)] = (char *) 0;
    i = read(fileno(stdin), line, MAXLINE);
    if (i > 0) {
        printchopped(HostName, line, i+1, fileno(stdin));
    } else if (i < 0) {
        if (errno != EINTR) {
            logerror("stdin");
        }
    }
    FD_CLR(fileno(stdin), &readfds);
}

#endif
}

int usage()
{
    fprintf(stderr, "usage: syslogd [-drvh] [-l hostlist] [-m markinterval] [-n] [-p path]\n" \
        "[-s domainlist] [-f conffile]\n");
    exit(1);
}

#ifdef SYSLOG_UNIXAF
static int create_unix_socket(const char *path)
{
    struct sockaddr_un sunx;
    int fd;
    int n;
    char line[MAXLINE + 1];

    n = sd_listen_fds();

```

```

if (n > 1) {
return -1;
} else if (n == 1) {
socket_from_systemd = 1;
fd = SD_LISTEN_FDS_START + 0;
} else {
if (path[0] == '\0')
return -1;

(void) unlink(path);

memset(&sunx, 0, sizeof(sunx));
sunx.sun_family = AF_UNIX;
(void) strncpy(sunx.sun_path, path, sizeof(sunx.sun_path));
fd = socket(AF_UNIX, SOCK_DGRAM, 0);
if (fd < 0 || bind(fd, (struct sockaddr *) &sunx,
    sizeof(sunx.sun_family)+strlen(sunx.sun_path)) < 0 ||
    chmod(path, 0666) < 0) {
(void) snprintf(line, sizeof(line), "cannot create %s", path);
logerror(line);
dprintf("cannot create %s (%d).\n", path, errno);
close(fd);
#ifdef SYSV
die(0);
#endif
return -1;
}
}
return fd;
}
#endif

#ifdef SYSLOG_INET
static int create_inet_socket()
{
int fd, on = 1;
struct sockaddr_in sin;
int sockflags;

fd = socket(AF_INET, SOCK_DGRAM, 0);
if (fd < 0) {
logerror("syslog: Unknown protocol, suspending inet service.");
return fd;
}

memset(&sin, 0, sizeof(sin));
sin.sin_family = AF_INET;
sin.sin_port = LogPort;
if (setsockopt(fd, SOL_SOCKET, SO_REUSEADDR, \
    (char *) &on, sizeof(on)) < 0) {
logerror("setsockopt(REUSEADDR), suspending inet");
close(fd);
return -1;
}
/* We must not block on the network socket, in case a packet

```

```

    * gets lost between select and recv, otherwise the process
    * will stall until the timeout, and other processes trying to
    * log will also stall.
    */
    if ((sockflags = fcntl(fd, F_GETFL)) != -1) {
        sockflags |= O_NONBLOCK;
    /*
     * SETFL could fail too, so get it caught by the subsequent
     * error check.
     */
        sockflags = fcntl(fd, F_SETFL, sockflags);
    }
    if (sockflags == -1) {
        logerror("fcntl(O_NONBLOCK), suspending inet");
        close(fd);
        return -1;
    }
    if (bind(fd, (struct sockaddr *) &sin, sizeof(sin)) < 0) {
        logerror("bind, suspending inet");
        close(fd);
        return -1;
    }
    return fd;
}
#endif

char **
crunch_list(list)
char *list;
{
    int i, m, n;
    char *p, *q;
    char **result = NULL;

    p = list;

    /* strip off trailing delimiters */
    while (*p && p[strlen(p)-1] == LIST_DELIMITER)
        p[strlen(p)-1] = '\0';
    /* cut off leading delimiters */
    while (p[0] == LIST_DELIMITER)
        p++;

    /* count delimiters to calculate the number of elements */
    for (n = i = 0; p[i]; i++)
        if (p[i] == LIST_DELIMITER) n++;

    if ((result = (char **) malloc(sizeof(char *) * (n + 2))) == NULL) {
        printf("Sorry, can't get enough memory, exiting.\n");
        exit(1);
    }

    /*
     * We now can assume that the first and last

```

```

    * characters are different from any delimiters,
    * so we don't have to care about this.
    */
    m = 0;
    while ((q = strchr(p, LIST_DELIMITER)) && m < n) {
        result[m] = (char *) malloc((q - p + 1) * sizeof(char));
        if (result[m] == NULL) {
            printf ("Sorry, can't get enough memory, exiting.\n");
            exit(1);
        }
        memcpy(result[m], p, q - p);
        result[m][q - p] = '\0';
        p = q; p++;
        m++;
    }
    if ((result[m] = strdup(p)) == NULL) {
        printf ("Sorry, can't get enough memory, exiting.\n");
        exit(1);
    }
    result[++m] = NULL;

#ifdef 0
    m = 0;
    while (result[m])
        dprintf ("#%d: %s\n", m, result[m++]);
#endif
    return result;
}

void unTTY()
#ifdef SYSV
{
    if ( !Debug ) {
        setsid();
    }
    return;
}

#else
{
    int i;

    if ( !Debug ) {
        i = open(_PATH_TTY, O_RDWR);
        if (i >= 0) {
            (void) ioctl(i, (int) TIOCNOTTY, (char *)0);
            (void) close(i);
        }
    }
}
#endif

/*

```

```

* Parse the line to make sure that the msg is not a composite of more
* than one message.
*/

```

```

void printchopped(hname, msg, len, fd)
const char *hname;
char *msg;
int len;
int fd;
{
    auto int ptlngth;

    auto char *start = msg,
        *p,
        *end,
        tmpline[MAXLINE + 1];

    dprintf("Message length: %d, File descriptor: %d.\n", len, fd);
    tmpline[0] = '\0';
    if ( parts[fd] != (char *) 0 )
    {
        dprintf("Including part from messages.\n");
        strcpy(tmpline, parts[fd]);
        free(parts[fd]);
        parts[fd] = (char *) 0;
        if ( (strlen(msg) + strlen(tmpline)) > MAXLINE )
        {
            logerror("Cannot glue message parts together");
            printline(hname, tmpline);
            start = msg;
        }
        else
        {
            dprintf("Previous: %s\n", tmpline);
            dprintf("Next: %s\n", msg);
            strcat(tmpline, msg); /* length checked above */
            printline(hname, tmpline);
            if ( (strlen(msg) + 1) == len )
                return;
            else
                start = strchr(msg, '\0') + 1;
        }
    }

    if ( msg[len-1] != '\0' )
    {
        msg[len] = '\0';
        for(p= msg+len-1; *p != '\0' && p > msg; )
            --p;
        if(*p == '\0') p++;
        ptlngth = strlen(p);
        if ( (parts[fd] = malloc(ptlngth + 1)) == (char *) 0 )
            logerror("Cannot allocate memory for message part.");
        else

```

```

{
strcpy(parts[fd], p);
dprintf("Saving partial msg: %s\n", parts[fd]);
memset(p, '\0', ptlength);
}
}

do {
end = strchr(start + 1, '\0');
printline(hname, start);
start = end + 1;
} while ( *start != '\0' );

return;
}

/*
 * Take a raw input line, decode the message, and print the message
 * on the appropriate log files.
 */

void printline(hname, msg)
const char *hname;
char *msg;
{
register char *p, *q;
register unsigned char c;
char line[MAXLINE + 1];
unsigned int pri;    // Valid Priority values are 0-191
int prilen=0; // Track Priority value string len
int msglen;

/* test for special codes */
msglen=strlen(msg);
pri = DEFUPRI;
p = msg;

if (*p == '<') {
pri = 0;
while (--msglen > 0 && isdigit((unsigned char)*++p) &&
pri <= MAX_PRI) {
pri = 10 * pri + (*p - '0');
prilen++;
}
if (*p == '>' && prilen)
++p;
else {
pri = DEFUPRI;
p = msg;
}
}

if ((pri &~ (LOG_FACMASK|LOG_PRIMASK)) || (pri > MAX_PRI)) {

```

```

    pri = DEFUPRI;
    p = msg;
}

memset (line, 0, sizeof(line));
q = line;
while ((c = *p++) && q < &line[sizeof(line) - 4]) {
    if (c == '\n' || c == 127)
        *q++ = ' ';
    else if (c < 040) {
        *q++ = '^';
        *q++ = c ^ 0100;
    } else
        *q++ = c;
    }
    *q = '\0';

    logmsg(pri, line, hname, SYNC_FILE);
    return;
}

/*
 * Take a raw input line from /dev/klog, split and format similar to syslog().
 */

void printsys(msg)
char *msg;
{
    register char *p, *q;
    register int c;
    char line[MAXLINE + 1];
    int pri, flags;
    char *lp;

    (void) snprintf(line, sizeof(line), "vmunix: ");
    lp = line + strlen(line);
    for (p = msg; *p != '\0'; ) {
        flags = ADDDATE;
        pri = DEFSPRI;
        if (*p == '<') {
            pri = 0;
            while (isdigit(*++p))
                pri = 10 * pri + (*p - '0');
            if (*p == '>')
                ++p;
        } else {
            /* kernel printf's come out on console */
            flags |= IGN_CONS;
        }
        if (pri &~ (LOG_FACMASK|LOG_PRIMASK))
            pri = DEFSPRI;
        q = lp;

```

```

while (*p != '\0' && (c = *p++) != '\n' &&
    q < &line[MAXLINE])
    *q++ = c;
*q = '\0';
logmsg(pri, line, LocalHostName, flags);
}
return;
}

/*
 * Decode a priority into textual information like auth.emerg.
 */
char *textpri(pri)
int pri;
{
    static char res[20];
    CODE *c_pri, *c_fac;

    for (c_fac = facilitynames; c_fac->c_name && !(c_fac->c_val == LOG_FAC(pri)<<3); c_fac++);
    for (c_pri = prioritynames; c_pri->c_name && !(c_pri->c_val == LOG_PRI(pri)); c_pri++);

    snprintf(res, sizeof(res), "%s.%s<%d>", c_fac->c_name, c_pri->c_name, pri);

    return res;
}

time_tnow;

/*
 * Log a message to the appropriate log files, users, etc. based on
 * the priority.
 */

void logmsg(pri, msg, from, flags)
int pri;
char *msg;
const char *from;
int flags;
{
    register struct filed *f;
    int fac, prilev, lognum;
    int msglen;
    char *timestamp;
#ifdef __gnu_linux__
    sigset_t mask;
#else
#ifdef SYSV
    sigset_t omask;
#endif
#endif

    dprintf("logmsg: %s, flags %x, from %s, msg %s\n", textpri(pri), flags, from, msg);

#ifdef __gnu_linux__
    sigemptyset(&mask);

```



```

sigaddset(&mask, SIGHUP);
sigaddset(&mask, SIGALRM);
sigprocmask(SIG_BLOCK, &mask, NULL);
#else
#ifdef SYSV
omask = sigblock(sigmask(SIGHUP)|sigmask(SIGALRM));
#endif
#endif

/*
 * Check to see if msg looks non-standard.
 *
 * A message looks like
 * Nov 17 11:42:33 CRON[
 * 01234567890123456
 * ^ ^ ^ ^ ^
 *
 * Remote messages are not accompanied by a timestamp.
 * Local messages are accompanied by a timestamp (program's timezone)
 */
msglen = strlen(msg);
if (!(msglen < 16 || msg[3] != ' ' || msg[6] != ' ' ||
    msg[9] != ':' || msg[12] != '.' || msg[15] != ' ')) {
    msg += 16;
    msglen -= 16;
}

(void) time(&now);
timestamp = ctime(&now) + 4;

/* extract facility and priority level */
fac = LOG_FAC(pri);
prilev = LOG_PRI(pri);

/* log the message to the particular outputs */
if (!Initialized) {
    f = &consfile;
    f->f_file = open(ctty, O_WRONLY|O_NOCTTY);

    if (f->f_file >= 0) {
        unttty();
        fprintflog(f, (char *)from, flags, msg);
        (void) close(f->f_file);
        f->f_file = -1;
    }
#ifdef __gnu_linux__
    sigprocmask(SIG_UNBLOCK, &mask, NULL);
#else
#ifdef SYSV
    (void) sigsetmask(omask);
#endif
#endif
    return;
}

```

```

#ifndef SYSV
for (lognum = 0; lognum <= nlogs; lognum++) {
f = &Files[lognum];
#else
for (f = Files; f; f = f->f_next) {
#endif

/* skip messages that are incorrect priority */
if ( (f->f_pmask[fac] == TABLE_NOPRI) || \
    ((f->f_pmask[fac] & (1<<prilev)) == 0) )
    continue;

if (f->f_type == F_CONSOLE && (flags & IGN_CONS))
    continue;

/* don't output marks to recently written files */
if ((flags & MARK) && (now - f->f_time) < MarkInterval / 2)
    continue;

/*
 * suppress duplicate lines to this file
 */
if ((flags & MARK) == 0 && msglen == f->f_prevlen &&
    !strcmp(msg, f->f_prevline) &&
    !strcmp(from, f->f_prevhost)) {
(void) strncpy(f->f_lasttime, timestamp, 15);
f->f_prevcount++;
dprintf("msg repeated %d times, %ld sec of %d.\n",
    f->f_prevcount, now - f->f_time,
    repeatinterval[f->f_repeatcount]);
/*
 * If domark would have logged this by now,
 * flush it now (so we don't hold isolated messages),
 * but back off so we'll flush less often
 * in the future.
 */
if (now > REPEATTIME(f)) {
fprintlog(f, (char *)from, flags, (char *)NULL);
BACKOFF(f);
}
} else {
/* new line, save it */
if (f->f_prevcount)
fprintlog(f, (char *)from, 0, (char *)NULL);
f->f_prevpri = pri;
f->f_repeatcount = 0;
(void) strncpy(f->f_lasttime, timestamp, 15);
(void) strncpy(f->f_prevhost, from,
sizeof(f->f_prevhost));
if (msglen < MAXSVLINE) {
f->f_prevlen = msglen;
(void) strcpy(f->f_prevline, msg);
fprintlog(f, (char *)from, flags, (char *)NULL);
} else {
f->f_prevline[0] = 0;

```

```

f->f_prevlen = 0;
fprintlog(f, (char *)from, flags, msg);
}
}
}
#ifdef __gnu_linux__
sigprocmask(SIG_UNBLOCK, &mask, NULL);
#else
#ifdef SYSV
(void) sigsetmask(omask);
#endif
#endif
}
#ifdef FALSE
} /* balance parentheses for emacs */
#endif

void fprintlog(f, from, flags, msg)
register struct filed *f;
char *from;
int flags;
char *msg;
{
struct iovec iov[6];
register struct iovec *v = iov;
char repbuf[80];
#ifdef SYSLOG_INET
register int l;
char line[MAXLINE + 1];
time_t fwd_suspend;
struct hostent *hp;
#endif

dprintf("Called fprintlog, ");

v->iov_base = f->f_lasttime;
v->iov_len = 15;
v++;
v->iov_base = " ";
v->iov_len = 1;
v++;
v->iov_base = f->f_prevhost;
v->iov_len = strlen(v->iov_base);
v++;
v->iov_base = " ";
v->iov_len = 1;
v++;
if (msg) {
v->iov_base = msg;
v->iov_len = strlen(msg);
} else if (f->f_prevcount > 1) {
(void) snprintf(repbuf, sizeof(repbuf), "last message repeated %d times",
f->f_prevcount);
v->iov_base = repbuf;

```

```

v->iov_len = strlen(repbuf);
} else {
v->iov_base = f->f_prevline;
v->iov_len = f->f_prevlen;
}
v++;

dprintf("logging to %s", TypeNameNames[f->f_type]);

switch (f->f_type) {
case F_UNUSED:
f->f_time = now;
dprintf("\n");
break;

#ifdef SYSLOG_INET
case F_FORW_SUSP:
fwd_suspend = time((time_t *) 0) - f->f_time;
if ( fwd_suspend >= INET_SUSPEND_TIME ) {
dprintf("\nForwarding suspension over, " \
"retrying FORW ");
f->f_type = F_FORW;
goto f_forw;
}
else {
dprintf(" %s\n", f->f_un.f_forw.f_hname);
dprintf("Forwarding suspension not over, time " \
"left: %d.\n", INET_SUSPEND_TIME - \
fwd_suspend);
}
break;

/*
 * The trick is to wait some time, then retry to get the
 * address. If that fails retry x times and then give up.
 *
 * You'll run into this problem mostly if the name server you
 * need for resolving the address is on the same machine, but
 * is started after syslogd.
 */
case F_FORW_UNKN:
dprintf(" %s\n", f->f_un.f_forw.f_hname);
fwd_suspend = time((time_t *) 0) - f->f_time;
if ( fwd_suspend >= INET_SUSPEND_TIME ) {
dprintf("Forwarding suspension to unknown over, retrying\n");
if ( (hp = gethostbyname(f->f_un.f_forw.f_hname)) == NULL ) {
dprintf("Failure: %s\n", sys_h_errlist[h_errno]);
dprintf("Retries: %d\n", f->f_prevcount);
if ( --f->f_prevcount < 0 ) {
dprintf("Giving up.\n");
f->f_type = F_UNUSED;
}
else
dprintf("Left retries: %d\n", f->f_prevcount);
}
}

```

```

    else {
        dprintf("%s found, resuming.\n", f->f_un.f_forw.f_hname);
        memcpy((char *) &f->f_un.f_forw.f_addr.sin_addr, hp->h_addr, hp->h_length);
        f->f_prevcount = 0;
        f->f_type = F_FORW;
        goto f_forw;
    }
    else
        dprintf("Forwarding suspension not over, time " \
            "left: %d\n", INET_SUSPEND_TIME - fwd_suspend);
        break;

case F_FORW:
/*
 * Don't send any message to a remote host if it
 * already comes from one. (we don't care 'bout who
 * sent the message, we don't send it anyway) -Joey
 */
f_forw:
dprintf(" %s\n", f->f_un.f_forw.f_hname);
if ( strcmp(from, LocalHostName) && NoHops )
dprintf("Not sending message to remote.\n");
else {
    f->f_time = now;
    (void) snprintf(line, sizeof(line), "<%d>%s", f->f_prevpri, \
        (char *) iov[4].iov_base);
    l = strlen(line);
    if (l > MAXLINE)
        l = MAXLINE;
    if (sendto(finnet, line, l, 0, \
        (struct sockaddr *) &f->f_un.f_forw.f_addr,
        sizeof(f->f_un.f_forw.f_addr)) != l) {
        int e = errno;
        dprintf("INET sendto error: %d = %s.\n",
            e, strerror(e));
        f->f_type = F_FORW_SUSP;
        errno = e;
        logerror("sendto");
    }
}
break;
#endif

case F_CONSOLE:
f->f_time = now;
#ifdef UNIXPC
if (1) {
#else
if (flags & IGN_CONS) {
#endif
dprintf(" (ignored).\n");
break;
}

```

```

/* FALLTHROUGH */

case F_TTY:
case F_FILE:
case F_PIPE:
f->f_time = now;
dprintf(" %s\n", f->f_un.f_fname);
if (f->f_type == F_TTY || f->f_type == F_CONSOLE) {
v->iov_base = "\r\n";
v->iov_len = 2;
} else {
v->iov_base = "\n";
v->iov_len = 1;
}
again:
/* f->f_file == -1 is an indicator that we couldn't
   open the file at startup. */
if (f->f_file == -1)
break;

if (writev(f->f_file, iov, 6) < 0) {
int e = errno;

/* If a named pipe is full, just ignore it for now */
if (f->f_type == F_PIPE && e == EAGAIN)
break;

/* If the filesystem is filled up, just ignore
   it for now and continue writing when
   possible */
if (f->f_type == F_FILE && e == ENOSPC)
break;

(void) close(f->f_file);
/*
 * Check for EBADF on TTY's due to vhangup() XXX
 * Linux uses EIO instead (mrn 12 May 96)
 */
if ((f->f_type == F_TTY || f->f_type == F_CONSOLE)
#ifdef linux
&& e == EIO) {
#else
&& e == EBADF) {
#endif
f->f_file = open(f->f_un.f_fname, O_WRONLY|O_APPEND|O_NOCTTY);
if (f->f_file < 0) {
f->f_type = F_UNUSED;
logerror(f->f_un.f_fname);
} else {
untty();
goto again;
}
} else {
f->f_type = F_UNUSED;
errno = e;

```

```

    logerror(f->f_un.f_fname);
}
} else if (f->f_type == F_FILE && (f->f_flags & SYNC_FILE))
(void) fsync(f->f_file);
break;

case F_USERS:
case F_WALL:
f->f_time = now;
dprintf("\n");
v->iov_base = "\r\n";
v->iov_len = 2;
wallmsg(f, iov);
break;
} /* switch */
if (f->f_type != F_FORW_UNKN)
f->f_prevcount = 0;
return;
}
#endif
} /* balance parentheses for emacs */
#endif

jmp_buf ttybuf;

void endtty()
{
longjmp(ttybuf, 1);
}

/*
 * WALLMSG -- Write a message to the world at large
 *
 * Write the specified message to either the entire
 * world, or a list of approved users.
 */

void wallmsg(f, iov)
register struct filed *f;
struct iovec *iov;
{
char p[sizeof(_PATH_DEV) + UNAMESZ];
register int i;
int ttyf, len;
static int reenter = 0;
struct utmp ut;
struct utmp *uptr;
char greetings[200];

(void) &len;

if (reenter++)
return;

```

```

/* open the user login file */
setutent();

/*
 * Might as well fork instead of using nonblocking I/O
 * and doing notty().
 */
if (fork() == 0) {
/* Trying to pass on the LISTEN_PID with appropriate value */
const char *e;
unsigned long l;
char buf[24] = { '\0' };

if ( (e = getenv("LISTEN_PID")) ) {
errno = 0;
l = strtoul(e, NULL, 10);
if (errno == 0) {
if (getppid() == (pid_t)l) {
snprintf(buf, sizeof(buf), "%d", (int)getpid());
setenv("LISTEN_PID", buf, 1);
}
}
}

(void) signal(SIGTERM, SIG_DFL);
(void) alarm(0);
#ifdef SYSV
(void) signal(SIGTTOU, SIG_IGN);
(void) sigsetmask(0);
#endif
(void) snprintf(greetings, sizeof(greetings),
    "\r\n\7Message from syslogd@%s at %.24s ...\r\n",
(char *) iov[2].iov_base, ctime(&now));
len = strlen(greetings);

/* scan the user login file */
while ((uptr = getutent())) {
memcpy(&ut, uptr, sizeof(ut));
/* is this slot used? */
if (ut.ut_name[0] == '\0')
continue;
if (ut.ut_type != USER_PROCESS)
continue;
if (!(strcmp (ut.ut_name, "LOGIN"))) /* paranoia */
continue;

/* should we send the message to this user? */
if (f->f_type == F_USERS) {
for (i = 0; i < MAXUNAMES; i++) {
if (!f->f_un.f_ufname[i][0]) {
i = MAXUNAMES;
break;
}
}
if (strncmp(f->f_un.f_ufname[i],
    ut.ut_name, UNAMESZ) == 0)

```



```
break;
}
if (i >= MAXUNAMES)
continue;
}

/* compute the device name */
strcpy(p, _PATH_DEV);
strncat(p, ut.ut_line, UNAMESZ);

if (f->f_type == F_WALL) {
iov[0].iov_base = greetings;
iov[0].iov_len = len;
iov[1].iov_len = 0;
}
if (setjmp(ttybuf) == 0) {
(void) signal(SIGALRM, endtty);
(void) alarm(15);
/* open the terminal */
ttyf = open(p, O_WRONLY|O_NOCTTY);
if (ttyf >= 0) {
struct stat statb;

if (fstat(ttyf, &statb) == 0 &&
    (statb.st_mode & S_IWRITE))
(void) writev(ttyf, iov, 6);
close(ttyf);
ttyf = -1;
}
}
(void) alarm(0);
}
exit(0);
}
/* close the user login file */
endutent();
reenter = 0;
}

void reapchild()
{
int saved_errno = errno;
#ifdef(SYSV) && !defined(linux)
(void) signal(SIGCHLD, reapchild);/* reset signal handler -ASP */
wait ((int *)0);
#else
int status;

while (wait3(&status, WNOHANG, (struct rusage *) NULL) > 0)
;
#endif
#ifdef linux
(void) signal(SIGCHLD, reapchild);/* reset signal handler -ASP */
#endif
}
```

```

    errno = saved_errno;
}

/*
 * Return a printable representation of a host address.
 */
const char *cvthname(f)
struct sockaddr_in *f;
{
    struct hostent *hp;
    register char *p;
    int count;

    if (f->sin_family != AF_INET) {
        dprintf("Malformed from address.\n");
        return ("???");
    }
    hp = gethostbyaddr((char *) &f->sin_addr, sizeof(struct in_addr), \
        f->sin_family);
    if (hp == 0) {
        dprintf("Host name for your address (%s) unknown.\n",
            inet_ntoa(f->sin_addr));
        return (inet_ntoa(f->sin_addr));
    }
    /*
     * Convert to lower case, just like LocalDomain above
     */
    for (p = (char *)hp->h_name; *p ; p++)
        if (isupper(*p))
            *p = tolower(*p);

    /*
     * Notice that the string still contains the fqdn, but your
     * hostname and domain are separated by a '\0'.
     */
    if ((p = strchr(hp->h_name, '.')) {
        if (strcmp(p + 1, LocalDomain) == 0) {
            *p = '\0';
            return (hp->h_name);
        } else {
            if (StripDomains) {
                count=0;
                while (StripDomains[count]) {
                    if (strcmp(p + 1, StripDomains[count]) == 0) {
                        *p = '\0';
                        return (hp->h_name);
                    }
                    count++;
                }
            }
            if (LocalHosts) {
                count=0;
                while (LocalHosts[count]) {
                    if (!strcmp(hp->h_name, LocalHosts[count])) {
                        *p = '\0';

```

```

return (hp->h_name);
}
count++;
}
}
}
}

return (hp->h_name);
}

void domark()
{
register struct filed *f;
#ifdef SYSV
int lognum;
#endif

if (MarkInterval > 0) {
now = time(0);
MarkSeq += TIMERINTVL;
if (MarkSeq >= MarkInterval) {
logmsg(LOG_MARK|LOG_INFO, "-- MARK --", LocalHostName, ADDDATE|MARK);
MarkSeq = 0;
}
}

#ifdef SYSV
for (lognum = 0; lognum <= nlogs; lognum++) {
f = &Files[lognum];
#else
for (f = Files; f; f = f->f_next) {
#endif
if (f->f_prevcount && now >= REPEATTIME(f)) {
dprintf("flush %s: repeated %d times, %d sec.\n",
TypeNames[f->f_type], f->f_prevcount,
repeatinterval[f->f_repeatcount]);
fprintlog(f, LocalHostName, 0, (char *)NULL);
BACKOFF(f);
}
}
}

(void) signal(SIGALRM, domark);
(void) alarm(TIMERINTVL);
}

void debug_switch()
{
dprintf("Switching debugging_on to %s\n", (debugging_on == 0) ? "true" : "false");
debugging_on = (debugging_on == 0) ? 1 : 0;
signal(SIGUSR1, debug_switch);
}

```

```

/*
 * Print syslogd errors some place.
 */
void logerror(type)
char *type;
{
char buf[100];

dprintf("Called logerr, msg: %s\n", type);

if (errno == 0)
(void) snprintf(buf, sizeof(buf), "syslogd: %s", type);
else
(void) snprintf(buf, sizeof(buf), "syslogd: %s: %s", type, strerror(errno));
errno = 0;
logmsg(LOG_SYSLOG|LOG_ERR, buf, LocalHostName, ADDDATE);
return;
}

void die(sig)

int sig;

{
register struct filed *f;
char buf[100];
int lognum;
int i;
int was_initialized = Initialized;

Initialized = 0; /* Don't log SIGCHLDs in case we
    receive one during exiting */

for (lognum = 0; lognum <= nlogs; lognum++) {
f = &Files[lognum];
/* flush any pending output */
if (f->f_prevcount)
fprintf(f, LocalHostName, 0, (char *)NULL);
}

Initialized = was_initialized;
if (sig) {
dprintf("syslogd: exiting on signal %d\n", sig);
(void) snprintf(buf, sizeof(buf), "exiting on signal %d", sig);
errno = 0;
logmsg(LOG_SYSLOG|LOG_INFO, buf, LocalHostName, ADDDATE);
}

/* Close the UNIX sockets. */
for (i = 0; i < nfunix; i++)
if (funix[i] != -1)
close(funix[i]);
/* Close the inet socket. */
if (InetInuse) close(inetm);

```

```
/* Clean-up files. */
if (socket_from_systemd) {
    i = 1;
} else {
    i = 0;
}
for (; i < nfunix; i++)
    if (funixn[i] && funix[i] != -1)
        (void)unlink(funixn[i]);

#ifdef TESTING
(void) remove_pid(PidFile);
#endif
exit(0);
}

/*
 * Signal handler to terminate the parent process.
 */
#ifdef TESTING
void doexit(sig)
int sig;
{
    exit (0);
}
#endif

/*
 * INIT -- Initialize syslogd from configuration table
 */

void init()
{
    register int i, lognum;
    register FILE *cf;
    register struct filed *f;
#ifdef TESTING
#ifdef SYSV
register struct filed **nextp = (struct filed **) 0;
#endif
#endif
    register char *p;
    register unsigned int Forwarding = 0;
#ifdef CONT_LINE
    char cbuf[BUFSIZ];
    char *cline;
#else
    char cline[BUFSIZ];
#endif
    struct servent *sp;

    sp = getservbyname("syslog", "udp");
    if (sp == NULL) {
        if (errno == ENOENT) {
```

```

    errno = 0;
    logerror("The file /etc/services does not seem exist.");
}
errno = 0;
logerror("network logging disabled (syslog/udp service unknown).");
logerror("see syslogd(8) for details of whether and how to enable it.");
LogPort = 0;
} else
LogPort = sp->s_port;

/*
 * Close all open log files and free log descriptor array.
 */
dprintf("Called init.\n");
Initialized = 0;
if ( nlogs > -1 )
{
    dprintf("Initializing log structures.\n");

    for (lognum = 0; lognum <= nlogs; lognum++) {
        f = &Files[lognum];

        /* flush any pending output */
        if (f->f_prevcount)
            fprintf(f, LocalHostName, 0, (char *)NULL);

        switch (f->f_type) {
            case F_FILE:
            case F_PIPE:
            case F_TTY:
            case F_CONSOLE:
                (void) close(f->f_file);
                break;
        }
    }

    /*
     * This is needed especially when HUPing syslogd as the
     * structure would grow infinitively. -Joey
     */
    nlogs = -1;
    free((void *) Files);
    Files = (struct filed *) 0;
}

#ifdef SYSV
lognum = 0;
#else
f = NULL;
#endif

/* open the configuration file */
if ((cf = fopen(ConfFile, "r")) == NULL) {
    dprintf("cannot open %s.\n", ConfFile);
}

```

```

#ifdef SYSV
allocate_log();
f = &Files[lognum++];
#endif TESTING
cfile("*.err\t" _PATH_CONSOLE, f);
#else
snprintf(cbuf, sizeof(cbuf), "%.t%s", ttyname(0));
cfile(cbuf, f);
#endif
#else
*nextp = (struct filed *)calloc(1, sizeof(*f));
cfile("*.ERR\t" _PATH_CONSOLE, *nextp);
(*nextp)->f_next = (struct filed *)calloc(1, sizeof(*f))/ ASP */
cfile("*.PANIC\t", (*nextp)->f_next);
#endif
Initialized = 1;
return;
}

/*
 * Foreach line in the conf table, open that file.
 */
#ifdef CONT_LINE
cline = cbuf;
while (fgets(cline, sizeof(cbuf) - (cline - cbuf), cf) != NULL) {
#else
while (fgets(cline, sizeof(cline), cf) != NULL) {
#endif
/*
 * check for end-of-section, comments, strip off trailing
 * spaces and newline character.
 */
for (p = cline; isspace(*p); ++p);
if (*p == '\0' || *p == '#')
continue;
#ifdef CONT_LINE
memmove(cline, p, strlen(p)+1);
#endif
for (p = strchr(cline, '\0'); isspace(*--p););
#ifdef CONT_LINE
if (*p == '\\') {
if ((p - cbuf) > BUFSIZ - 30) {
/* Oops the buffer is full - what now? */
cline = cbuf;
} else {
*p = 0;
cline = p;
continue;
}
} else
cline = cbuf;
#endif
*++p = '\0';
#endif SYSV

```

```

f = (struct filed *)calloc(1, sizeof(*f));
*nextp = f;
nextp = &f->f_next;
#endif
allocate_log();
f = &Files[lognum++];
#if CONT_LINE
cflne(cbuf, f);
#else
cflne(cline, f);
#endif
if (f->f_type == F_FORW || f->f_type == F_FORW_SUSP || f->f_type == F_FORW_UNKN) {
Forwarding++;
}
}

/* close the configuration file */
(void) fclose(cf);

#ifndef SYSLOG_UNIXAF
for (i = 0; i < nfunix; i++) {
if (funix[i] != -1)
/* Don't close the socket, preserve it instead
close(funix[i]);
*/
continue;
if ((funix[i] = create_unix_socket(funixn[i])) != -1)
dprintf("Opened UNIX socket '%s'.\n", funixn[i]);
}
#endif

#ifndef SYSLOG_INET
if (Forwarding || AcceptRemote) {
if (finet < 0) {
finet = create_inet_socket();
if (finet >= 0) {
InetInuse = 1;
dprintf("Opened syslog UDP port.\n");
}
}
}
else {
if (finet >= 0)
close(finet);
finet = -1;
InetInuse = 0;
}
inetm = finet;
#endif

Initialized = 1;

if ( Debug ) {
#ifdef SYSV
for (lognum = 0; lognum <= nlogs; lognum++) {

```



```

f = &Files[lognum];
if (f->f_type != F_UNUSED) {
printf ("%2d: ", lognum);
#else
for (f = Files; f; f = f->f_next) {
if (f->f_type != F_UNUSED) {
#endif
for (i = 0; i <= LOG_NFACILITIES; i++)
if (f->f_pmask[i] == TABLE_NOPRI)
printf(" X ");
else
printf("%2X ", f->f_pmask[i]);
printf("%s: ", TypeName[f->f_type]);
switch (f->f_type) {
case F_FILE:
case F_PIPE:
case F_TTY:
case F_CONSOLE:
printf("%s", f->f_un.f_fname);
if (f->f_file == -1)
printf(" (unused)");
break;

case F_FORW:
case F_FORW_SUSP:
case F_FORW_UNKN:
printf("%s", f->f_un.f_forw.f_hname);
break;

case F_USERS:
for (i = 0; i < MAXUNAMES && *f->f_un.f_uname[i]; i++)
printf("%s, ", f->f_un.f_uname[i]);
break;
}
printf("\n");
}
}
}

if ( AcceptRemote )
#ifdef DEBRELEASe
logmsg(LOG_SYSLOG|LOG_INFO, "syslogd " VERSION "." PATCHLEVEL "#" DEBRELEASe \
": restart (remote reception).", LocalHostName, \
ADDDATE);
#else
logmsg(LOG_SYSLOG|LOG_INFO, "syslogd " VERSION "." PATCHLEVEL \
": restart (remote reception).", LocalHostName, \
ADDDATE);
#endif
else
#ifdef DEBRELEASe
logmsg(LOG_SYSLOG|LOG_INFO, "syslogd " VERSION "." PATCHLEVEL "#" DEBRELEASe \
": restart.", LocalHostName, ADDDATE);
#else

```

```

logmsg(LOG_SYSLOG|LOG_INFO, "syslogd " VERSION "." PATCHLEVEL \
      ": restart." , LocalHostName, ADDDATE);
#endif
(void) signal(SIGHUP, sighup_handler);
dprintf("syslogd: restarted.\n");
}
#if FALSE
}}} /* balance parentheses for emacs */
#endif

/*
 * Crack a configuration file line
 */

void cfline(line, f)
char *line;
register struct filed *f;
{
register char *p;
register char *q;
register int i, i2;
char *bp;
int pri;
int singlpri = 0;
int ignorepri = 0;
int syncfile;
#ifdef SYSLOG_INET
struct hostent *hp;
#endif
char buf[MAXLINE];
char xbuf[200];

dprintf("cfline(%s)\n", line);

errno = 0; /* keep strerror() stuff out of logerror messages */

/* clear out file entry */
#ifdef SYSV
memset((char *) f, 0, sizeof(*f));
#endif
for (i = 0; i <= LOG_NFACILITIES; i++) {
f->f_pmask[i] = TABLE_NOPRI;
f->f_flags = 0;
}

/* scan through the list of selectors */
for (p = line; *p && *p != '\t' && *p != ' ';) {

/* find the end of this facility name list */
for (q = p; *q && *q != '\t' && *q++ != ' '; )
continue;

/* collect priority name */
for (bp = buf; *q && !strchr("\t ,;", *q); )
*bp++ = *q++;

```

```

*bp = '\0';

/* skip cruft */
while (strchr(".,", *q))
q++;

/* decode priority name */
if ( *buf == '!' ) {
ignorepri = 1;
for (bp=buf; *(bp+1); bp++)
*bp=*(bp+1);
*bp='\0';
}
else {
ignorepri = 0;
}
if ( *buf == '=' )
{
singlpri = 1;
pri = decode(&buf[1], PriNames);
}
else {
singlpri = 0;
pri = decode(buf, PriNames);
}

if (pri < 0) {
(void) snprintf(xbuf, sizeof(xbuf), "unknown priority name \"%s\"", buf);
logerror(xbuf);
return;
}

/* scan facilities */
while (*p && !strchr("\t .;", *p)) {
for (bp = buf; *p && !strchr("\t .;", *p); )
*bp++ = *p++;
*bp = '\0';
if (*buf == '*') {
for (i = 0; i <= LOG_NFACILITIES; i++) {
if ( pri == INTERNAL_NOPRI ) {
if ( ignorepri )
f->f_pmask[i] = TABLE_ALLPRI;
else
f->f_pmask[i] = TABLE_NOPRI;
}
else if ( singlpri ) {
if ( ignorepri )
f->f_pmask[i] &= ~(1<<pri);
else
f->f_pmask[i] |= (1<<pri);
}
}
else
{
if ( pri == TABLE_ALLPRI ) {

```

```

    if ( ignorepri )
    f->f_pmask[i] = TABLE_NOPRI;
    else
    f->f_pmask[i] = TABLE_ALLPRI;
    }
    else
    {
    if ( ignorepri )
    for (i2= 0; i2 <= pri; ++i2)
    f->f_pmask[i] &= ~(1<<i2);
    else
    for (i2= 0; i2 <= pri; ++i2)
    f->f_pmask[i] |= (1<<i2);
    }
    }
    }
    } else {
    i = decode(buf, FacNames);
    if (i < 0) {

    (void) snprintf(xbuf, sizeof(xbuf), "unknown facility name \"%s\"", buf);
    logerror(xbuf);
    return;
    }

    if ( pri == INTERNAL_NOPRI ) {
    if ( ignorepri )
    f->f_pmask[i >> 3] = TABLE_ALLPRI;
    else
    f->f_pmask[i >> 3] = TABLE_NOPRI;
    } else if ( singlpri ) {
    if ( ignorepri )
    f->f_pmask[i >> 3] &= ~(1<<pri);
    else
    f->f_pmask[i >> 3] |= (1<<pri);
    } else {
    if ( pri == TABLE_ALLPRI ) {
    if ( ignorepri )
    f->f_pmask[i >> 3] = TABLE_NOPRI;
    else
    f->f_pmask[i >> 3] = TABLE_ALLPRI;
    } else {
    if ( ignorepri )
    for (i2= 0; i2 <= pri; ++i2)
    f->f_pmask[i >> 3] &= ~(1<<i2);
    else
    for (i2= 0; i2 <= pri; ++i2)
    f->f_pmask[i >> 3] |= (1<<i2);
    }
    }
    }
    while (*p == ';' || *p == ' ')
    p++;
    }

```

```

p = q;
}

/* skip to action part */
while (*p == '\t' || *p == ' ')
p++;

if (*p == '-')
{
syncfile = 0;
p++;
} else
syncfile = 1;

dprintf("leading char in action: %c\n", *p);
switch (*p)
{
case '@':
#ifdef SYSLOG_INET
if (!LogPort) {
f->f_type = F_UNUSED;
logerror("Forward rule without networking enabled");
break;
}

(void) strcpy(f->f_un.f_forw.f_hname, ++p);
dprintf("forwarding host: %s\n", p); /*ASP*/
if ( (hp = gethostbyname(p)) == NULL ) {
f->f_type = F_FORW_UNKN;
f->f_prevcount = INET_RETRY_MAX;
f->f_time = time ( (time_t *)0 );
} else {
f->f_type = F_FORW;
}

memset((char *) &f->f_un.f_forw.f_addr, 0,
sizeof(f->f_un.f_forw.f_addr));
f->f_un.f_forw.f_addr.sin_family = AF_INET;
f->f_un.f_forw.f_addr.sin_port = LogPort;
if ( f->f_type == F_FORW )
memcpy((char *) &f->f_un.f_forw.f_addr.sin_addr, hp->h_addr, hp->h_length);
/*
 * Otherwise the host might be unknown due to an
 * inaccessible nameserver (perhaps on the same
 * host). We try to get the ip number later, like
 * FORW_SUSP.
 */
#endif
break;

case '|':
case '/':
(void) strcpy(f->f_un.f_fname, p);
dprintf ("filename: %s\n", p); /*ASP*/

```

```

if (syncfile)
f->f_flags |= SYNC_FILE;
if ( *p == '|' ) {
f->f_file = open(++p, O_RDWR|O_NONBLOCK|O_NOCTTY);
f->f_type = F_PIPE;
} else {
f->f_file = open(p, O_WRONLY|O_APPEND|O_CREAT|O_NONBLOCK|O_NOCTTY,
0644);
f->f_type = F_FILE;
}

if ( f->f_file < 0 ){
f->f_file = -1;
dprintf("Error opening log file: %s\n", p);
logerror(p);
break;
}
if (isatty(f->f_file)) {
f->f_type = F_TTY;
untty();
}
if (strcmp(p, cty) == 0)
f->f_type = F_CONSOLE;
break;

case '*':
dprintf("write-all\n");
f->f_type = F_WALL;
break;

default:
dprintf("users: %s\n", p);/* ASP */
for (i = 0; i < MAXUNAMES && *p; i++) {
for (q = p; *q && *q != ','; )
q++;
(void) strncpy(f->f_un.f_uname[i], p, UNAMESZ);
if ((q - p) > UNAMESZ)
f->f_un.f_uname[i][UNAMESZ] = '\0';
else
f->f_un.f_uname[i][q - p] = '\0';
while (*q == ',' || *q == ' ')
q++;
p = q;
}
f->f_type = F_USERS;
break;
}
return;
}

/*
* Decode a symbolic name to a numeric value
*/

```

```
int decode(name, codetab)
char *name;
struct code *codetab;
{
register struct code *c;
register char *p;
char buf[80];

dprintf ("symbolic name: %s", name);
if (isdigit(*name))
{
dprintf ("\n");
return (atoi(name));
}
(void) strncpy(buf, name, 79);
for (p = buf; *p; p++)
if (isupper(*p))
*p = tolower(*p);
for (c = codetab; c->c_name; c++)
if (!strcmp(buf, c->c_name))
{
dprintf ("==> %d\n", c->c_val);
return (c->c_val);
}
return (-1);
}

static void dprintf(char *fmt, ...)

{
va_list ap;

if ( !(Debug && debugging_on) )
return;

va_start(ap, fmt);
vfprintf(stdout, fmt, ap);
va_end(ap);

fflush(stdout);
return;
}

/*
 * The following function is responsible for allocating/reallocating the
 * array which holds the structures which define the logging outputs.
 */
static void allocate_log()

{
dprintf("Called allocate_log, nlogs = %d.\n", nlogs);

/*
```

```

    * Decide whether the array needs to be initialized or needs to
    * grow.
    */
    if ( nlogs == -1 )
    {
        Files = (struct filed *) malloc(sizeof(struct filed));
        if ( Files == (void *) 0 )
        {
            dprintf("Cannot initialize log structure.");
            logerror("Cannot initialize log structure.");
            return;
        }
    }
    else
    {
        /* Re-allocate the array. */
        Files = (struct filed *) realloc(Files, (nlogs+2) * \
            sizeof(struct filed));
        if ( Files == (struct filed *) 0 )
        {
            dprintf("Cannot grow log structure.");
            logerror("Cannot grow log structure.");
            return;
        }
    }

    /*
    * Initialize the array element, bump the number of elements in the
    * the array and return.
    */
    ++nlogs;
    memset(&Files[nlogs], '0', sizeof(struct filed));
    return;
}

/*
* The following function is responsible for handling a SIGHUP signal. Since
* we are now doing mallocs/free as part of init we had better not being
* doing this during a signal handler. Instead this function simply sets
* a flag variable which will tell the main loop to go through a restart.
*/
void sighup_handler()

{
    restart = 1;
    signal(SIGHUP, sighup_handler);
    return;
}

int sd_listen_fds() {
    int r, fd;
    const char *e;
    unsigned long l;

```



```
e = getenv("LISTEN_PID");
if(!e) {
return 0;
}

errno = 0;
l = strtoul(e, NULL, 10);

if (errno != 0) {
return -errno;
}

if (getpid() != (pid_t)l) {
return 0;
}

e = getenv("LISTEN_FDS");
if(!e) {
return 0;
}

errno = 0;
l = strtoul(e, NULL, 10);

if (errno != 0) {
return -errno;
}

for (fd = SD_LISTEN_FDS_START; fd < SD_LISTEN_FDS_START + (int)l; fd++) {
int flags;

flags = fcntl(fd, F_GETFD);
if (flags < 0) {
return -errno;
}

if (flags & FD_CLOEXEC)
continue;

if (fcntl(fd, F_SETFD, flags | FD_CLOEXEC) < 0) {
return -errno;
}
}

r = (int)l;

return r;
}
/*
 * Local variables:
 * c-indent-level: 8
 * c-basic-offset: 8
 * tab-width: 8
 * End:
 */
```

1.131 sysstat

1.131.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc., <<http://fsf.org/>>
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
{description}
Copyright (C) {year} {fullname}
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

{signature of Ty Coon}, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.132 sysvinit

1.132.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Sysvinit is Copyright (C) 1991-2004 Miquel van Smoorenburg

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

On Debian GNU/Linux systems, the complete text of the GNU General Public License can be found in `/usr/share/common-licenses/GPL-2'.

Send patches to sysvinit-devel@nongnu.org

The of the start-stop-daemon

- * A rewrite of the original Debian's start-stop-daemon Perl script
- * in C (faster - it is executed many times during system startup).
- *
- * Written by Marek Michalkiewicz <marekm@i17linuxb.ists.pwr.wroc.pl>,
- * public domain.

1.133 tar

1.133.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received. You must make sure that they, too, receive
or can get the source code. And you must show them these terms so they
know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software. For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated

conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the

Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read [<http://www.gnu.org/philosophy/why-not-lgpl.html>](http://www.gnu.org/philosophy/why-not-lgpl.html).

1.134 tcl

1.134.1 Available under license

This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation, ActiveState Corporation and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7014 (b) (3) of DFARs. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

1.135 tcp-wrappers

1.135.1 Available under license

```
/*
 * Copyright 1995 by Wietse Venema. All rights reserved. Some individual
 * files may be covered by other copyrights.
 *
 * This material was originally written and compiled by Wietse Venema at
 * Eindhoven University of Technology, The Netherlands, in 1990, 1991,
 * 1992, 1993, 1994 and 1995.
 *
 * Redistribution and use in source and binary forms are permitted
 * provided that this entire copyright notice is duplicated in all such
 * copies.
 *
 * This software is provided "as is" and without any expressed or implied
 * warranties, including, without limitation, the implied warranties of
 * merchantability and fitness for any particular purpose.
 */
```

1.136 tcpdump

1.136.1 Available under license

License: BSD

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

1.137 time

1.137.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) 19yy <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

1.138 tnftp

1.138.1 Available under license

\$NetBSD: COPYING,v 1.7 2010/01/12 07:01:01 lukem Exp \$

Copyright (c) 2001-2010 The NetBSD Foundation, Inc.
All rights reserved.

This code is derived from software contributed to The NetBSD Foundation
by Simon Burge, Luke Mewburn, and Christos Zoulas.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software
must display the following acknowledgement:
This product includes software developed by the NetBSD
Foundation, Inc. and its contributors.
4. Neither the name of The NetBSD Foundation nor the names of its
contributors may be used to endorse or promote products derived
from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS
``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS
BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

This product also contains software developed by other people, and you
are advised to read the various source files to read the full details
of the other licenses. Those licenses also require the following
acknowledgements:

This product includes software developed by the NetBSD Foundation,
Inc. and its contributors. Those contributors include:

- Simon Burge
- Jaromir Dolecek
- Klaus Klein

- Luke Mewburn
- Jason R. Thorpe of the Numerical Aerospace Simulation Facility, NASA Ames Research Center.
- Christos Zoulas

This product includes software developed by the University of California, Berkeley and its contributors. Those contributors include:

- Christos Zoulas of Cornell University.
- Guido van Rossum.

This product includes software developed by Christos Zoulas.

1.139 traceroute

1.139.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original

authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source

code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS

TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid

distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes

a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from

such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a

derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more

than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and

all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.140 unzip

1.140.1 Available under license

This is version 2009-Jan-02 of the Info-ZIP license.

The definitive version of this document should be available at <ftp://ftp.info-zip.org/pub/infozip/license.html> indefinitely and a copy at <http://www.info-zip.org/pub/infozip/license.html>.

Copyright (c) 1990-2009 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda, Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the above disclaimer and the following restrictions:

1. Redistributions of source code (in whole or in part) must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2. Redistributions in binary form (compiled executables and libraries) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. Additional documentation is not needed for executables where a command line license option provides these and a note regarding this option is in the executable's startup banner. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.
3. Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, versions with modified or added functionality, and dynamic, shared, or static library

versions not from Info-ZIP--must be plainly marked as such and must not be misrepresented as being the original source or, if binaries, compiled from the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but not limited to, labeling of the altered versions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP will provide support for the altered versions.

4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

1.141 update-rc.d

1.141.1 Available under license

```
#!/bin/sh
#
# update-rc.d Update the links in /etc/rc[0-9S].d/
#
# (c) 2003, 2004 Phil Blundell <pb@handhelds.org>
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2, or (at your option)
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.

initd="/etc/init.d"
etcd="/etc/rc"
notreally=0
force=0
dostart=0
verbose=0

usage()
{
cat >&2 <<EOF
usage: update-rc.d [-n] [-f] [-r <root>] <basename> remove
      update-rc.d [-n] [-r <root>] [-s] <basename> defaults [NN | sNN kNN]
      update-rc.d [-n] [-r <root>] [-s] <basename> start|stop NN runlvl [runlvl] [...] .
-n: not really
-f: force
-v: verbose
-r: alternate root path (default is /)
-s: invoke start methods if appropriate to current runlevel
EOF
}

checklinks()
{
local i dn fn remove=0
if [ "x$1" = "xremove" ]; then
echo " Removing any system startup links for $bn ..."
remove=1
fi

for i in 0 1 2 3 4 5 6 7 8 9 S; do
dn="${etcd}/${i}.d"
```

```

if [ ! -d $dn ]; then
continue;
fi
for f in ${dn}/[SK]??${bn}; do
if [ -L $f ]; then
if [ $remove -eq 0 ]; then
return 1
fi
echo " $f"
if [ $notreally -eq 1 ]; then
continue
fi
rm $f
fi
done
done

return 0
}

dolink()
{
startstop=$1
lev=`echo $2 | cut -d/ -f1`
nn=`echo $2 | cut -d/ -f2`
fn="${etcd}${lev}.d/${startstop}${nn}${bn}"
[ $verbose -eq 1 ] && echo " $fn -> ../init.d/${bn}"
if [ $notreally -eq 0 ]; then
mkdir -p `dirname $fn`
ln -s ../init.d/${bn} $fn
fi
if [ $dostart -eq 1 ] && [ $startstop = "S" ] && [ $lev = $RUNLEVEL ]; then
$fn start || true
fi
}

makelinks()
{
if ! checklinks; then
echo " System startup links for $initd/$bn already exist."
if [ $dostart -eq 1 ] && [ $notreally -eq 0 ] && [ -L ${etcd}${RUNLEVEL}.d/S??${bn} ]; then
${etcd}${RUNLEVEL}.d/S??${bn} restart || true
fi
exit 0
fi

echo " Adding system startup for $initd/$bn."

for i in $startlinks; do
dolink S $i
done
for i in $stoplinks; do
dolink K $i
done
}

```

```
while [ $# -gt 0 ]; do
case $1 in
-n)notreally=1
shift
continue
;;
-v)verbose=1
shift
continue
;;
-f)force=1
shift
continue
;;
-s)dostart=1
shift
continue
;;
-r) shift
root=$1
initd="${root}/${initd}"
etcd="${root}/${etcd}"
shift
;;
-h | --help)
usage
exit 0
;;
-*)
usage
exit 1
;;
*)
break
;;
esac
done

if [ $# -lt 2 ]; then
usage
exit 1
fi

bn=$1
shift

sn=$initd/$bn
if [ -L "$sn" -a -n "$root" ]; then
if which readlink >/dev/null; then
while true; do
linksn="$(readlink "$sn")"
if [ -z "$linksn" ]; then
break
```

```

fi

sn="$linksn"
case "$sn" in
/*) sn="$root$sn" ;;
*) sn="$initd/$sn" ;;
esac
done
else
echo "update-rc.d: readlink tool not present, cannot check whether \
$sn symlink points to a valid file." >&2
fi
fi

if [ $1 != "remove" ]; then
if [ ! -f "$sn" ]; then
echo "update-rc.d: $initd/$bn: file does not exist" >&2
exit 1
fi
else
if [ -f "$sn" ]; then
if [ $force -eq 1 ]; then
echo "update-rc.d: $initd/$bn exists during rc.d purge (continuing)" >&2
else
echo "update-rc.d: $initd/$bn exists during rc.d purge (use -f to force)" >&2
exit 1
fi
fi
fi

if [ $dostart -eq 1 ]; then
#RUNLEVEL=`sed 's/.*\[(.*)\]/1/' < /proc/1/cmdline`
RUNLEVEL=`runlevel | cut -d" " -f2`
if [ "x$RUNLEVEL" = "x" ]; then
echo "Unable to determine current runlevel" >&2
exit 1
fi
fi

case $1 in
remove)
checklinks "remove"
;;

defaults)
if [ $# -gt 3 ]; then
echo "defaults takes only one or two arguments" >&2
usage
exit 1
fi
start=20
stop=20
if [ $# -gt 1 ]; then
start=$2
stop=$2

```

```

fi
if [ $# -gt 2 ]; then
stop=$3
fi
start=`printf %02d $start`
stop=`printf %02d $stop`
stoplinks="0/$stop 1/$stop 6/$stop"
startlinks="2/$start 3/$start 4/$start 5/$start"
makelinks
;;

start | stop)
while [ $# -gt 0 ]; do
if [ $1 = "start" ]; then
letter=S
elif [ $1 = "stop" ]; then
letter=K
else
echo "expected start or stop" >&2
usage
exit 1
fi
shift
NN=`printf %02d $(expr $1 + 0)`
shift
while [ "x$1" != "x." ]; do
if [ $# -eq 0 ]; then
echo "action with list of runlevels not terminated by \`.`" >&2
exit 1
fi
level=$1
shift
case $letter in
S) startlinks="$startlinks $level/$NN" ;;
K) stoplinks="$stoplinks $level/$NN" ;;
esac
done
shift
done
makelinks
;;

*)
usage
exit 1
;;
esac

```

1.142 usbutils

1.142.1 Available under license

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any

associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to

apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate

parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

1.143 util-linux

1.143.1 Available under license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, and the entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ALL OF WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF NOT ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you

have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under

the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying

the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence

the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or

other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form

under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one

of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise

permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any

such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

/*

* Copyright (c) 1989 The Regents of the University of California.

* All rights reserved.

*

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:

* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.

* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.

* 3. All advertising materials mentioning features or use of this software
* must display the following acknowledgement:

*This product includes software developed by the University of
*California, Berkeley and its contributors.

* 4. Neither the name of the University nor the names of its contributors
* may be used to endorse or promote products derived from this software
* without specific prior written permission.

*

* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.

*/

The project util-linux doesn't use the same license for all of the code.

There is code under:

* GPLv2+ (GNU General Public License version 2, or any later version)

* GPLv2 (GNU General Public License version 2)

* LGPLv2+ (GNU Lesser General Public License v2 (or 2.1) or any later version)

* BSD with advertising

* Public Domain

Please, check the source code for more details. A license is usually at the start
of each source file.

The ./COPYING file (GPLv2+) is the default license for code without an explicitly
defined license.

1.144 vim-tiny

1.144.1 Available under license

uganda.txt For Vim version 8.0. Last change: 2013 Jul 06

VIM REFERENCE MANUAL by Bram Moolenaar

uganda *Uganda* *copying* *copyright* *license*

SUMMARY

iccf *ICCF*

Vim is Charityware. You can use and copy it as much as you like, but you are encouraged to make a donation for needy children in Uganda. Please see |kcc| below or visit the ICCF web site, available at these URLs:

<http://iccf-holland.org/>
<http://www.vim.org/iccf/>
<http://www.iccf.nl/>

You can also sponsor the development of Vim. Vim sponsors can vote for features. See |sponsor|. The money goes to Uganda anyway.

The Open Publication License applies to the Vim documentation, see |manual-copyright|.

==== begin of license ====

VIM LICENSE

- I) There are no restrictions on distributing unmodified copies of Vim except that they must include this license text. You can also distribute unmodified parts of Vim, likewise unrestricted except that they must include this license text. You are also allowed to include executables that you made from the unmodified Vim sources, plus your own usage examples and Vim scripts.
- II) It is allowed to distribute a modified (or extended) version of Vim, including executables and/or source code, when the following four conditions are met:
 - 1) This license text must be included unmodified.
 - 2) The modified Vim must be distributed in one of the following five ways:
 - a) If you make changes to Vim yourself, you must clearly describe in the distribution how to contact you. When the maintainer asks you (in any way) for a copy of the modified Vim you distributed, you must make your changes, including source code, available to the maintainer without fee. The maintainer reserves the right to include your changes in the official version of Vim. What the maintainer will do with your changes and under what license they will be distributed is negotiable. If there has been no negotiation then this license, or a later version, also applies to your changes.

The current maintainer is Bram Moolenaar <Bram@vim.org>. If this changes it will be announced in appropriate places (most likely vim.sf.net, www.vim.org and/or comp.editors). When it is completely impossible to contact the maintainer, the obligation to send him your changes ceases. Once the maintainer has confirmed that he has received your changes they will not have to be sent again.

b) If you have received a modified Vim that was distributed as mentioned under a) you are allowed to further distribute it unmodified, as mentioned at I). If you make additional changes the text under a) applies to those changes.

c) Provide all the changes, including source code, with every copy of the modified Vim you distribute. This may be done in the form of a context diff. You can choose what license to use for new code you add. The changes and their license must not restrict others from making their own changes to the official version of Vim.

d) When you have a modified Vim which includes changes as mentioned under c), you can distribute it without the source code for the changes if the following three conditions are met:

- The license that applies to the changes permits you to distribute the changes to the Vim maintainer without fee or restriction, and permits the Vim maintainer to include the changes in the official version of Vim without fee or restriction.
- You keep the changes for at least three years after last distributing the corresponding modified Vim. When the maintainer or someone who you distributed the modified Vim to asks you (in any way) for the changes within this period, you must make them available to him.
- You clearly describe in the distribution how to contact you. This contact information must remain valid for at least three years after last distributing the corresponding modified Vim, or as long as possible.

e) When the GNU General Public License (GPL) applies to the changes, you can distribute the modified Vim under the GNU GPL version 2 or any later version.

3) A message must be added, at least in the output of the ":version" command and in the intro screen, such that the user of the modified Vim is able to see that it was modified. When distributing as mentioned under 2)e) adding the message is only required for as far as this does not conflict with the license used for the changes.

4) The contact information as required under 2)a) and 2)d) must not be removed or changed, except that the person himself can make corrections.

III) If you distribute a modified version of Vim, you are encouraged to use the Vim license for your changes and make them available to the maintainer, including the source code. The preferred way to do this is by e-mail or by uploading the files to a server and e-mailing the URL. If the number of changes is small (e.g., a modified Makefile) e-mailing a context diff will do. The e-mail address to be used is <maintainer@vim.org>

IV) It is not allowed to remove this license from the distribution of the Vim sources, parts of it or from a modified version. You may use this license for previous Vim releases instead of the license that they came

with, at your option.

==== end of license ====

Note:

- If you are happy with Vim, please express that by reading the rest of this file and consider helping needy children in Uganda.
- If you want to support further Vim development consider becoming a [sponsor]. The money goes to Uganda anyway.
- According to Richard Stallman the Vim license is GNU GPL compatible. A few minor changes have been made since he checked it, but that should not make a difference.
- If you link Vim with a library that goes under the GNU GPL, this limits further distribution to the GNU GPL. Also when you didn't actually change anything in Vim.
- Once a change is included that goes under the GNU GPL, this forces all further changes to also be made under the GNU GPL or a compatible license.
- If you distribute a modified version of Vim, you can include your name and contact information with the "--with-modified-by" configure argument or the MODIFIED_BY define.

Kibaale Children's Centre*kcc* *Kibaale* *charity*

Kibaale Children's Centre (KCC) is located in Kibaale, a small town in the south of Uganda, near Tanzania, in East Africa. The area is known as Rakai District. The population is mostly farmers. Although people are poor, there is enough food. But this district is suffering from AIDS more than any other part of the world. Some say that it started there. Estimations are that 10 to 30% of the Ugandans are infected with HIV. Because parents die, there are many orphans. In this district about 60,000 children have lost one or both parents, out of a population of 350,000. And this is still continuing.

The children need a lot of help. The KCC is working hard to provide the needy with food, medical care and education. Food and medical care to keep them healthy now, and education so that they can take care of themselves in the future. KCC works on a Christian base, but help is given to children of any religion.

The key to solving the problems in this area is education. This has been neglected in the past years with president Idi Amin and the following civil wars. Now that the government is stable again, the children and parents have to learn how to take care of themselves and how to avoid infections. There is also help for people who are ill and hungry, but the primary goal is to prevent people from getting ill and to teach them how to grow healthy food.

Most of the orphans are living in an extended family. An uncle or older sister is taking care of them. Because these families are big and the income (if any) is low, a child is lucky if it gets healthy food. Clothes, medical

care and schooling is beyond its reach. To help these needy children, a sponsorship program was put into place. A child can be financially adopted. For a few dollars a month KCC sees to it that the child gets indispensable items, is healthy, goes to school and KCC takes care of anything else that needs to be done for the child and the family that supports it.

Besides helping the child directly, the environment where the child grows up needs to be improved. KCC helps schools to improve their teaching methods. There is a demonstration school at the centre and teacher trainings are given. Health workers are being trained, hygiene education is carried out and households are stimulated to build a proper latrine. I helped setting up a production site for cement slabs. These are used to build a good latrine. They are sold below cost price.

There is a small clinic at the project, which provides children and their family with medical help. When needed, transport to a hospital is offered. Immunization programs are carried out and help is provided when an epidemic is breaking out (measles and cholera have been a problem).

donate

Summer 1994 to summer 1995 I spent a whole year at the centre, working as a volunteer. I have helped to expand the centre and worked in the area of water and sanitation. I learned that the help that the KCC provides really helps. When I came back to Holland, I wanted to continue supporting KCC. To do this I'm raising funds and organizing the sponsorship program. Please consider one of these possibilities:

1. Sponsor a child in primary school: 17 euro a month (or more).
2. Sponsor a child in secondary school: 25 euro a month (or more).
3. Sponsor the clinic: Any amount a month or quarter
4. A one-time donation

Compared with other organizations that do child sponsorship the amounts are very low. This is because the money goes directly to the centre. Less than 5% is used for administration. This is possible because this is a small organization that works with volunteers. If you would like to sponsor a child, you should have the intention to do this for at least one year.

How do you know that the money will be spent right? First of all you have my personal guarantee as the author of Vim. I trust the people that are working at the centre, I know them personally. Furthermore, the centre has been co-sponsored and inspected by World Vision, Save the Children Fund and is now under the supervision of Pacific Academy Outreach Society. The centre is visited about once a year to check the progress (at our own cost). I have visited the centre myself many times, starting in 1993. The visit reports are on the ICCF web site.

If you have any further questions, send me e-mail: <Bram@vim.org>.

The address of the centre is:
Kibaale Children's Centre
p.o. box 1658
Masaka, Uganda, East Africa

Sending money: *iccf-donations*

Check the ICCF web site for the latest information! See [iccf] for the URL.

USA: The methods mentioned below can be used.
 Sending a check to the Nehemiah Group Outreach Society (NGOS) is no longer possible, unfortunately. We are looking for another way to get you an IRS tax receipt.
 For sponsoring a child contact KCF in Canada (see below). US checks can be sent to them to lower banking costs.

Canada: Contact Kibaale Children's Fund (KCF) in Surrey, Canada. They take care of the Canadian sponsors for the children in Kibaale. KCF forwards 100% of the money to the project in Uganda. You can send them a one time donation directly. Please send me a note so that I know what has been donated because of Vim. Ask KCF for information about sponsorship.
 Kibaale Children's Fund c/o Pacific Academy
 10238-168 Street
 Surrey, B.C. V4N 1Z4
 Canada
 Phone: 604-581-5353
 If you make a donation to Kibaale Children's Fund (KCF) you will receive a tax receipt which can be submitted with your tax return.

Holland: Transfer to the account of "Stichting ICCF Holland" in Lisse. This will allow for tax deduction if you live in Holland.
 Postbank, nr. 4548774
 IBAN: NL95 INGB 0004 5487 74

Germany: It is possible to make donations that allow for a tax return. Check the ICCF web site for the latest information:
<http://iccf-holland.org/germany.html>

World: Use a postal money order. That should be possible from any country, mostly from the post office. Use this name (which is in my passport): "Abraham Moolenaar". Use Euro for the currency if possible.

Europe: Use a bank transfer if possible. Your bank should have a form that you can use for this. See "Others" below for the swift code and IBAN number.
 Any other method should work. Ask for information about sponsorship.

Credit Card: You can use PayPal to send money with a Credit card. This is the most widely used Internet based payment system. It's really simple to use. Use this link to find more info:
https://www.paypal.com/en_US/mrb/pal=XAC62PML3GF8Q
 The e-mail address for sending the money to is:
 Bram@iccf-holland.org
 For amounts above 400 Euro (\$500) sending a check is preferred.

Others: Transfer to one of these accounts if possible:

Postbank, account 4548774

Swift code: INGB NL 2A

IBAN: NL95 INGB 0004 5487 74

under the name "stichting ICCF Holland", Lisse

If that doesn't work:

Rabobank Lisse, account 3765.05.117

Swift code: RABO NL 2U

under the name "Bram Moolenaar", Lisse

Otherwise, send a check in euro or US dollars to the address below. Minimal amount: \$70 (my bank does not accept smaller amounts for foreign check, sorry)

Address to send checks to:

Bram Moolenaar

Finsterruetihof 1

8134 Adliswil

Switzerland

This address is expected to be valid for a long time.

vim:tw=78:ts=8:ft=help:norl:

1.145 vlan

1.145.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder say-

ing it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more

than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geograph-

ical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License

as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision`
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

```
//
//Copyright (C) 2001 Ben Greear
//
//This program is free software; you can redistribute it and/or
//modify it under the terms of the GNU Library General Public License
//as published by the Free Software Foundation; either version 2
//of the License, or (at your option) any later version.
//
//This program is distributed in the hope that it will be useful,
//but WITHOUT ANY WARRANTY; without even the implied warranty of
//MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
//GNU General Public License for more details.
//
//You should have received a copy of the GNU Library General Public License
//along with this program; if not, write to the Free Software
//Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
//
```

```

// To contact the Author, Ben Greear: greearb@candelatech.com
//

#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <strings.h>
#include <sys/ioctl.h>
#include <linux/if_vlan.h>
#include <linux/sockios.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>

#define MAX_HOSTNAME 256

static char* usage =
    "\n"
    "Usage: add      [interface-name] [vlan_id]\n"
    "      rem      [vlan-name]\n"
    "      set_flag  [interface-name] [flag-num]  [0 | 1]\n"
    "      set_egress_map [vlan-name]  [skb_priority] [vlan_qos]\n"
    "      set_ingress_map [vlan-name]  [skb_priority] [vlan_qos]\n"
    "      set_name_type [name-type]\n"
    "\n"
    "* The [interface-name] is the name of the ethernet card that hosts\n"
    "  the VLAN you are talking about.\n"
    "* The vlan_id is the identifier (0-4095) of the VLAN you are operating on.\n"
    "* skb_priority is the priority in the socket buffer (sk_buff).\n"
    "* vlan_qos is the 3 bit priority in the VLAN header\n"
    "* name-type: VLAN_PLUS_VID (vlan0005), VLAN_PLUS_VID_NO_PAD (vlan5),\n"
    "             DEV_PLUS_VID (eth0.0005), DEV_PLUS_VID_NO_PAD (eth0.5)\n"
    "* bind-type: PER_DEVICE # Allows vlan 5 on eth0 and eth1 to be unique.\n"
    "             PER_KERNEL # Forces vlan 5 to be unique across all devices.\n"
    "* FLAGS: 1 REORDER_HDR When this is set, the VLAN device will move the\n"
    "           ethernet header around to make it look exactly like a real\n"
    "           ethernet device. This may help programs such as DHCPd which\n"
    "           read the raw ethernet packet and make assumptions about the\n"
    "           location of bytes. If you don't need it, don't turn it on, because\n"
    "           there will be at least a small performance degradation. Default\n"
    "           is OFF.\n";

void show_usage() {
    fprintf(stdout,usage);
}

int hex_to_bytes(char* bytes, int bytes_length, char* hex_str) {
    int hlen;
    int i;

```

```

int j = 0;
char hex[3];
char* stop; /* not used for any real purpose */

hlen = strlen(hex_str);

hex[2] = 0;

for (i = 0; i < hlen; i++) {

    hex[0] = hex_str[i];
    i++;
    if (i >= hlen) {
        return j; /* done */
    }

    hex[1] = hex_str[i];
    bytes[j++] = (char)strtoul(hex, &stop, 16);
}
return j;
}

int main(int argc, char** argv) {
    int fd;
    struct vlan_ioctl_args if_request;

    char* cmd = NULL;
    char* if_name = NULL;
    unsigned int vid = 0;
    unsigned int skb_priority;
    unsigned short vlan_qos;
    unsigned int nm_type = VLAN_NAME_TYPE_PLUS_VID;

    char* conf_file_name = "/proc/net/vlan/config";

    memset(&if_request, 0, sizeof(struct vlan_ioctl_args));

    if ((argc < 3) || (argc > 5)) {
        fprintf(stdout, "Expecting argc to be 3-5, inclusive. Was: %d\n", argc);

        show_usage();
        exit(1);
    }
    else {
        cmd = argv[1];

        if (strcasecmp(cmd, "set_name_type") == 0) {
            if (strcasecmp(argv[2], "VLAN_PLUS_VID") == 0) {
                nm_type = VLAN_NAME_TYPE_PLUS_VID;
            }
            else if (strcasecmp(argv[2], "VLAN_PLUS_VID_NO_PAD") == 0) {
                nm_type = VLAN_NAME_TYPE_PLUS_VID_NO_PAD;
            }
        }
    }
}

```

```

        else if (strcasemp(argv[2], "DEV_PLUS_VID") == 0) {
            nm_type = VLAN_NAME_TYPE_RAW_PLUS_VID;
        }
        else if (strcasemp(argv[2], "DEV_PLUS_VID_NO_PAD") == 0) {
            nm_type = VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD;
        }
        else {
            // MATHIEU
            //cerr << "Invalid name type.\n";
            fprintf(stderr, "Invalid name type.\n");

            show_usage();
            exit(1);
        }
        if_request.u.name_type = nm_type;
    }
    else {
        if_name = argv[2];
        if (strlen(if_name) > 15) {
            // MATHIEU
            //cerr << "ERROR: if_name must be 15 characters or less." << endl;
            fprintf(stderr, "ERROR: if_name must be 15 characters or less.\n");
            exit(1);
        }
        strcpy(if_request.device1, if_name);
    }

    if (argc == 4) {
        vid = atoi(argv[3]);
        if_request.u.VID = vid;
    }

    if (argc == 5) {
        skb_priority = atoi(argv[3]);
        vlan_qos = atoi(argv[4]);
        if_request.u.skp_priority = skb_priority;
        if_request.vlan_qos = vlan_qos;
    }
}

// Open up the /proc/vlan/config
if ((fd = open(conf_file_name, O_RDONLY)) < 0) {
    // MATHIEU
    //cerr << "ERROR: Could not open /proc/vlan/config.\n";
    fprintf(stderr, "WARNING: Could not open /proc/net/vlan/config. Maybe you need to load the 8021q module,
or maybe you are not using PROCFS??\n");

}
else {
    close(fd);
}

/* We use sockets now, instead of the file descriptor */
if ((fd = socket(AF_INET, SOCK_STREAM, 0)) < 0) {

```

```

    fprintf(stderr, "FATAL: Couldn't open a socket..go figure!\n");
    exit(2);
}

/* add */
if (strcasecmp(cmd, "add") == 0) {
    if_request.cmd = ADD_VLAN_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to add VLAN #%u to IF -:%s:- error: %s\n",
            vid, if_name, strerror(errno));
        exit(3);
    }
    else {
        fprintf(stdout, "Added VLAN with VID == %u to IF -:%s:-\n",
            vid, if_name);
        if (vid == 1) {
            fprintf(stdout, "WARNING: VLAN 1 does not work with many switches,\nconsider another number if you
have problems.\n");
        }
    }
} //if
else if (strcasecmp(cmd, "rem") == 0) {
    if_request.cmd = DEL_VLAN_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to remove VLAN -:%s:- error: %s\n",
            if_name, strerror(errno));
        exit(4);
    }
    else {
        fprintf(stdout, "Removed VLAN -:%s:-\n", if_name);
    }
} //if
else if (strcasecmp(cmd, "set_egress_map") == 0) {
    if_request.cmd = SET_VLAN_EGRESS_PRIORITY_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to set egress map on device -:%s:- error: %s\n",
            if_name, strerror(errno));
        exit(5);
    }
    else {
        fprintf(stdout, "Set egress mapping on device -:%s:- "
            "Should be visible in /proc/net/vlan/%s\n",
            if_name, if_name);
    }
}
else if (strcasecmp(cmd, "set_ingress_map") == 0) {
    if_request.cmd = SET_VLAN_INGRESS_PRIORITY_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to set ingress map on device -:%s:- error: %s\n",
            if_name, strerror(errno));
        exit(6);
    }
    else {
        fprintf(stdout, "Set ingress mapping on device -:%s:- "
            "Should be visible in /proc/net/vlan/%s\n",

```

```
        if_name, if_name);
    }
}
else if (strcasecmp(cmd, "set_flag") == 0) {
    if_request.cmd = SET_VLAN_FLAG_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to set flag on device -:%s:- error: %s\n",
            if_name, strerror(errno));
        exit(7);
    }
    else {
        fprintf(stdout, "Set flag on device -:%s:- "
            "Should be visible in /proc/net/vlan/%s\n",
            if_name, if_name);
    }
}
else if (strcasecmp(cmd, "set_name_type") == 0) {
    if_request.cmd = SET_VLAN_NAME_TYPE_CMD;
    if (ioctl(fd, SIOCSIFVLAN, &if_request) < 0) {
        fprintf(stderr, "ERROR: trying to set name type for VLAN subsystem, error: %s\n",
            strerror(errno));
        exit(8);
    }
    else {
        fprintf(stdout, "Set name-type for VLAN subsystem."
            " Should be visible in /proc/net/vlan/config\n");
    }
}
else {
    fprintf(stderr, "Unknown command -:%s:-\n", cmd);

    show_usage();
    exit(5);
}

return 0;
} /* main */
```

1.146 vsftpd

1.146.1 Available under license

vsftpd is licensed under version 2 of the GNU GPL.

As copyright holder, I give permission for vsftpd to be linked to the OpenSSL libraries. This includes permission for vsftpd binaries to be distributed linked against the OpenSSL libraries. All other obligations under the GPL v2 remain intact.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you

received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by

all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY

FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Everything within this tar archive is Copyright (c) Chris Evans, except where otherwise noted in individual files.

See "COPYING".

vsftpd is licensed under version 2 of the GNU GPL.
As copyright holder, I give permission for vsftpd to be linked to the OpenSSL libraries. This includes permission for vsftpd binaries to be distributed linked against the OpenSSL libraries. All other obligations under the GPL v2 remain intact.

1.147 watchdog-config

1.147.1 Available under license

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.148 watchdog

1.148.1 Available under license

Copyright (C) 1996-2012 Michael Meskes

WATCHDOG is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

WATCHDOG is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

1.149 which

1.149.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to

authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the

terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this

conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the

User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant

patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <<http://www.gnu.org/licenses/>>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <<http://www.gnu.org/philosophy/why-not-lgpl.html>>.

```
/*
 * which v2.x -- print full path of executables
 * Copyright (C) 1999, 2003, 2007, 2008 Carlo Wood <carlo@gnu.org>
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 */
```

```
#include "sys.h"
#include <stdio.h>
#include <ctype.h>
#include "getopt.h"
#include "tilde/tilde.h"
#include "bash.h"
```

```
static const char *prognam;
```

```

static void print_usage(FILE *out)
{
    fprintf(out, "Usage: %s [options] [--] COMMAND [...]\n", progname);
    fprintf(out, "Write the full path of COMMAND(s) to standard output.\n\n");
    fprintf(out, "  --version, -[vV] Print version and exit successfully.\n");
    fprintf(out, "  --help,          Print this help and exit successfully.\n");
    fprintf(out, "  --skip-dot       Skip directories in PATH that start with a dot.\n");
    fprintf(out, "  --skip-tilde     Skip directories in PATH that start with a tilde.\n");
    fprintf(out, "  --show-dot       Don't expand a dot to current directory in output.\n");
    fprintf(out, "  --show-tilde     Output a tilde for HOME directory for non-root.\n");
    fprintf(out, "  --tty-only       Stop processing options on the right if not on tty.\n");
    fprintf(out, "  --all, -a        Print all matches in PATH, not just the first\n");
    fprintf(out, "  --read-alias, -i Read list of aliases from stdin.\n");
    fprintf(out, "  --skip-alias     Ignore option --read-alias; don't read stdin.\n");
    fprintf(out, "  --read-functions Read shell functions from stdin.\n");
    fprintf(out, "  --skip-functions Ignore option --read-functions; don't read stdin.\n");
    fprintf(out, "Recommended use is to write the output of (alias; declare -f) to standard\n");
    fprintf(out, "input, so that which can show aliases and shell functions. See which(1) for\n");
    fprintf(out, "examples.\n\n");
    fprintf(out, "If the options --read-alias and/or --read-functions are specified then the\n");
    fprintf(out, "output can be a full alias or function definition, optionally followed by\n");
    fprintf(out, "the full path of each command used inside of those.\n\n");
    fprintf(out, "Report bugs to <which-bugs@gnu.org>.\n");
}

static void print_version(void)
{
    fprintf(stdout, "GNU which v" VERSION " , Copyright (C) 1999 - 2015 Carlo Wood.\n");
    fprintf(stdout, "GNU which comes with ABSOLUTELY NO WARRANTY;\n");
    fprintf(stdout, "This program is free software; your freedom to use, change\n");
    fprintf(stdout, "and distribute this program is protected by the GPL.\n");
}

static void print_fail(const char *name, const char *path_list)
{
    fprintf(stderr, "%s: no %s in (%s)\n", progname, name, path_list);
}

static char home[256];
static size_t homelen = 0;

static int absolute_path_given;
static int found_path_starts_with_dot;
static char *abs_path;

static int skip_dot = 0, skip_tilde = 0, skip_alias = 0, read_alias = 0;
static int show_dot = 0, show_tilde = 0, show_all = 0, tty_only = 0;
static int skip_functions = 0, read_functions = 0;

static char *find_command_in_path(const char *name, const char *path_list, int *path_index)
{
    char *found = NULL, *full_path;
    int status, name_len;

    name_len = strlen(name);

```

```
if (!absolute_program(name))
    absolute_path_given = 0;
else
{
    char *p;
    absolute_path_given = 1;

    if (abs_path)
        free(abs_path);

    if (*name != '.' && *name != '/' && *name != '~')
    {
        abs_path = (char *)xmalloc(3 + name_len);
        strcpy(abs_path, "./");
        strcat(abs_path, name);
    }
    else
    {
        abs_path = (char *)xmalloc(1 + name_len);
        strcpy(abs_path, name);
    }

    path_list = abs_path;
    p = strrchr(abs_path, '/');
    *p++ = 0;
    name = p;
}

while (path_list && path_list[*path_index])
{
    char *path;

    if (absolute_path_given)
    {
        path = savestring(path_list);
        *path_index = strlen(path);
    }
    else
        path = get_next_path_element(path_list, path_index);

    if (!path)
        break;

    if (*path == '~')
    {
        char *t = tilde_expand(path);
        free(path);
        path = t;

        if (skip_tilde)
        {
            free(path);
            continue;
        }
    }
}
```

```

    }
}

if (skip_dot && *path != '/')
{
    free(path);
    continue;
}

found_path_starts_with_dot = (*path == '.');

full_path = make_full_pathname(path, name, name_len);
free(path);

status = file_status(full_path);

if ((status & FS_EXISTS) && (status & FS_EXECABLE))
{
    found = full_path;
    break;
}

free(full_path);
}

return (found);
}

static char cwd[256];
static size_t cwdlen;

static void get_current_working_directory(void)
{
    if (cwdlen)
        return;

    if (!getcwd(cwd, sizeof(cwd)))
    {
        const char *pwd = getenv("PWD");
        if (pwd && strlen(pwd) < sizeof(cwd))
            strcpy(cwd, pwd);
    }

    if (*cwd != '/')
    {
        fprintf(stderr, "Can't get current working directory\n");
        exit(-1);
    }

    cwdlen = strlen(cwd);

    if (cwd[cwdlen - 1] != '/')
    {
        cwd[cwdlen++] = '/';
        cwd[cwdlen] = 0;
    }
}

```



```

    }
}

static char *path_clean_up(const char *path)
{
    static char result[256];

    const char *p1 = path;
    char *p2 = result;

    int saw_slash = 0, saw_slash_dot = 0, saw_slash_dot_dot = 0;

    if (*p1 != '/')
    {
        get_current_working_directory();
        strcpy(result, cwd);
        saw_slash = 1;
        p2 = &result[cwdlen];
    }

    do
    {
        /*
         * Two leading slashes are allowed, having an OS implementation-defined meaning.
         * See http://www.opengroup.org/onlinepubs/009695399/basedefs/xbd\_chap04.html#tag\_04\_11
         */
        if (!saw_slash || *p1 != '/' || (p1 == path + 1 && p1[1] != '/'))
            *p2++ = *p1;
        if (saw_slash_dot && (*p1 == '/'))
            p2 -= 2;
        if (saw_slash_dot_dot && (*p1 == '/'))
        {
            int cnt = 0;
            do
            {
                if (--p2 < result)
                {
                    strcpy(result, path);
                    return result;
                }
            } while (*p2 == '/')
            ++cnt;
            while (cnt != 3);
            ++p2;
        }
        saw_slash_dot_dot = saw_slash_dot && (*p1 == '.');
        saw_slash_dot = saw_slash && (*p1 == '.');
        saw_slash = (*p1 == '/');
    } while (*p1++);

    return result;
}

```

```

struct function_st {
    char *name;
    size_t len;
    char **lines;
    int line_count;
};

static struct function_st *functions;
static int func_count;
static int max_func_count;

static char **aliases;
static int alias_count;
static int max_alias_count;

int func_search(int indent, const char *cmd, struct function_st *func_list, int function_start_type)
{
    int i;
    for (i = 0; i < func_count; ++i)
    {
        if (!strcmp(functions[i].name, cmd))
        {
            int j;
            if (indent)
                fputc('\t', stdout);
            if (function_start_type == 1)
                fprintf(stdout, "%s () {\n", cmd);
            else
                fprintf(stdout, "%s ()\n", cmd);
            for (j = 0; j < functions[i].line_count; ++j)
            {
                if (indent)
                    fputc('\t', stdout);
                fputs(functions[i].lines[j], stdout);
            }
            return 1;
        }
    }
    return 0;
}

int path_search(int indent, const char *cmd, const char *path_list)
{
    char *result = NULL;
    int found_something = 0;

    if (path_list && *path_list != '\0')
    {
        int next;
        int path_index = 0;
        do
        {
            next = show_all;
            result = find_command_in_path(cmd, path_list, &path_index);
        }
    }
}

```

```

        if (result)
        {
            const char *full_path = path_clean_up(result);
            int in_home = (show_tilde || skip_tilde) && !strcmp(full_path, home, homelen);
            if (indent)
                fprintf(stdout, "\t");
            if (!(skip_tilde && in_home) && show_dot && found_path_starts_with_dot && !strcmp(full_path, cwd,
            cwdlen))
            {
                full_path += cwdlen;
                fprintf(stdout, "/");
            }
            else if (in_home)
            {
                if (skip_tilde)
                {
                    next = 1;
                    free(result);
                    continue;
                }
                if (show_tilde)
                {
                    full_path += homelen;
                    fprintf(stdout, "~/");
                }
            }
        }
        fprintf(stdout, "%s\n", full_path);
        free(result);
        found_something = 1;
    }
    else
        break;
    }
    while (next);
}

return found_something;
}

void process_alias(const char *str, int argc, char *argv[], const char *path_list, int function_start_type)
{
    const char *p = str;
    int len = 0;

    while(*p == ' ' || *p == '\t')
        ++p;
    if (!strcmp("alias", p, 5))
        p += 5;
    while(*p == ' ' || *p == '\t')
        ++p;
    while(*p && *p != ' ' && *p != '\t' && *p != '=')
        ++p, ++len;

    for (; argc > 0; --argc, ++argv)

```

```

{
    char q = 0;
    char *cmd;

    if (!*argv || len != strlen(*argv) || strncmp(*argv, &p[-len], len))
        continue;

    fputs(str, stdout);

    if (!show_all)
        *argv = NULL;

    while(*p == ' ' || *p == '\t')
        ++p;
    if (*p == '=')
        ++p;
    while(*p == ' ' || *p == '\t')
        ++p;
    if (*p == '"' || *p == '\n')
        q = *p, ++p;

    for(;;)
    {
        int found = 0;

        while(*p == ' ' || *p == '\t')
            ++p;
        len = 0;
        while(*p && *p != ' ' && *p != '\t' && *p != q && *p != '|' && *p != '&')
            ++p, ++len;

        cmd = (char *)xmalloc(len + 1);
        strncpy(cmd, &p[-len], len);
        cmd[len] = 0;
        if (*argv && !strcmp(cmd, *argv))
            *argv = NULL;
        if (read_functions && !strchr(cmd, '/'))
            found = func_search(1, cmd, functions, function_start_type);
        if (show_all || !found)
            path_search(1, cmd, path_list);
        free(cmd);

        while(*p && (*p != '|' || p[1] == '|') && (*p != '&' || p[1] == '&'))
            ++p;

        if (!*p)
            break;

        ++p;
    }

    break;
}
}

```

```
enum opts {
    opt_version,
    opt_skip_dot,
    opt_skip_tilde,
    opt_skip_alias,
    opt_read_functions,
    opt_skip_functions,
    opt_show_dot,
    opt_show_tilde,
    opt_tty_only,
    opt_help
};

#ifdef __TANDEM
/* According to Tom Bates, <tom.bates@hp.com> */
static uid_t const superuser = 65535;
#else
static uid_t const superuser = 0;
#endif

int main(int argc, char *argv[])
{
    const char *path_list = getenv("PATH");
    int short_option, fail_count = 0;
    static int long_option;
    struct option longopts[] = {
        {"help", 0, &long_option, opt_help},
        {"version", 0, &long_option, opt_version},
        {"skip-dot", 0, &long_option, opt_skip_dot},
        {"skip-tilde", 0, &long_option, opt_skip_tilde},
        {"show-dot", 0, &long_option, opt_show_dot},
        {"show-tilde", 0, &long_option, opt_show_tilde},
        {"tty-only", 0, &long_option, opt_tty_only},
        {"all", 0, NULL, 'a'},
        {"read-alias", 0, NULL, 'i'},
        {"skip-alias", 0, &long_option, opt_skip_alias},
        {"read-functions", 0, &long_option, opt_read_functions},
        {"skip-functions", 0, &long_option, opt_skip_functions},
        {NULL, 0, NULL, 0}
    };

    progname = argv[0];
    while ((short_option = getopt_long(argc, argv, "aivV", longopts, NULL)) != -1)
    {
        switch (short_option)
        {
            case 0:
                switch (long_option)
                {
                    case opt_help:
                        print_usage(stdout);
                        return 0;
                    case opt_version:
                        print_version();
```

```

    return 0;
case opt_skip_dot:
    skip_dot = !tty_only;
    break;
case opt_skip_tilde:
    skip_tilde = !tty_only;
    break;
case opt_skip_alias:
    skip_alias = 1;
    break;
case opt_show_dot:
    show_dot = !tty_only;
    break;
case opt_show_tilde:
    show_tilde = (!tty_only && geteuid() != superuser);
    break;
case opt_tty_only:
    tty_only = !isatty(1);
    break;
case opt_read_functions:
    read_functions = 1;
    break;
case opt_skip_functions:
    skip_functions = 1;
    break;
}
break;
    case 'a':
show_all = 1;
break;
    case 'i':
    read_alias = 1;
break;
    case 'v':
    case 'V':
print_version();
return 0;
    }
}

uidget();

if (show_dot)
    get_current_working_directory();

if (show_tilde || skip_tilde)
{
    const char *h;

    if (!(h = getenv("HOME")))
        h = sh_get_home_dir();

    strncpy(home, h, sizeof(home));
    home[sizeof(home) - 1] = 0;
    homelen = strlen(home);

```

```
    if (home[homelen - 1] != '/' && homelen < sizeof(home) - 1)
    {
        strcat(home, "/");
        ++homelen;
    }
}

if (skip_alias)
    read_alias = 0;

if (skip_functions)
    read_functions = 0;

argv += optind;
argc -= optind;

if (argc == 0)
{
    print_usage(stderr);
    return -1;
}

int function_start_type = 0;
if (read_alias || read_functions)
{
    char buf[1024];
    int processing_aliases = read_alias;

    if (isatty(0))
    {
        fprintf(stderr, "%s: %s: Warning: stdin is a tty.\n", progname,
            (read_functions ? read_alias ? "--read-functions, --read-alias, -i" : "--read-functions" : "--read-alias, -i"));
    }

    while (fgets(buf, sizeof(buf), stdin))
    {
        int looks_like_function_start = 0;
        int function_start_has_declare;
        if (read_functions)
        {
            // bash version 2.0.5a and older output a pattern for `str' like
            // declare -fx FUNCTION_NAME ()
            // {
            //   body
            // }
            //
            // bash version 2.0.5b and later output a pattern for `str' like
            // FUNCTION_NAME ()
            // {
            //   body
            // }
            char *p = buf + strlen(buf) - 1;
            while (isspace(*p) && p > buf + 2)
                --p;
```

```

if (*p == ') && p[-1] == '(' && p[-2] == ' ')
{
    looks_like_function_start = 1;
    function_start_has_declare = (strncmp("declare -", buf, 9) == 0);
}
// Add some zsh support here.
// zsh does output a pattern for `str` like
// FUNCTION () {
//   body
// }
if (p > buf + 4 && *p == '{' && p[-1] == ' ' &&
    p[-2] == ')' && p[-3] == '(' && p[-4] == ' ')
{
    looks_like_function_start = 1;
    function_start_type = 1;
    function_start_has_declare = 0;
}
    }
    if (processing_aliases && !looks_like_function_start)
    {
// bash version 2.0.5b can throw in lines like "declare -fx FUNCTION_NAME", eat them.
if (!strncmp("declare -", buf, 9))
    continue;
if (alias_count == max_alias_count)
{
    max_alias_count += 32;
    aliases = (char **)xrealloc(aliases, max_alias_count * sizeof(char *));
}
aliases[alias_count++] = strcpy((char *)xmalloc(strlen(buf) + 1), buf);
    }
    else if (read_functions && looks_like_function_start)
    {
        struct function_st *function;
        int max_line_count;

const char *p = buf;
int len = 0;

        processing_aliases = 0;

// Eat "declare -fx " at start of bash version 2.0.5a and older, if present.
if (function_start_has_declare)
{
    p += 9;
    while(*p && *p++ != ' ');
}

while(*p && *p != ' ')
    ++p, ++len;

if (func_count == max_func_count)
{
    max_func_count += 16;
    functions = (struct function_st *)xrealloc(functions, max_func_count * sizeof(struct function_st));
}

```



```

function = &functions[func_count++];
function->name = (char *)xmalloc(len + 1);
strncpy(function->name, &p[-len], len);
function->name[len] = 0;
function->len = len;
max_line_count = 32;
function->lines = (char **)xmalloc(max_line_count * sizeof(char *));
function->line_count = 0;
while (fgets(buf, sizeof(buf), stdin))
{
    size_t blen = strlen(buf);
    function->lines[function->line_count++] = strcpy((char *)xmalloc(blen + 1), buf);
    if (!strcmp(buf, "{}\n"))
        break;
    if (function->line_count == max_line_count)
    {
        max_line_count += 32;
        function->lines = (char **)xrealloc(function->lines, max_line_count * sizeof(char *));
    }
}
}
if (read_alias)
{
    int i;
    for (i = 0; i < alias_count; ++i)
        process_alias(alias[i], argc, argv, path_list, function_start_type);
}

for (; argc > 0; --argc, ++argv)
{
    int found_something = 0;

    if (!*argv)
        continue;

    if (read_functions && !strchr(*argv, '/'))
        found_something = func_search(0, *argv, functions, function_start_type);

    if ((show_all || !found_something) && !path_search(0, *argv, path_list) && !found_something)
    {
        print_fail(absolute_path_given ? strchr(*argv, '/') + 1 : *argv, absolute_path_given ? abs_path : path_list);
        ++fail_count;
    }
}

return fail_count;
}

#ifdef NEED_XMALLOC
void *xmalloc(size_t size)
{
    void *ptr = malloc(size);

```

```
    if (ptr == NULL)
    {
        fprintf(stderr, "%s: Out of memory", progname);
        exit(-1);
    }
    return ptr;
}

void *xrealloc(void *ptr, size_t size)
{
    if (!ptr)
        return xmalloc(size);
    ptr = realloc(ptr, size);
    if (size > 0 && ptr == NULL)
    {
        fprintf(stderr, "%s: Out of memory\n", progname);
        exit(-1);
    }
    return ptr;
}
#endif /* NEED_XMALLOC */
```

1.150 xinetd

1.150.1 Available under license

ORIGINAL LICENSE:

This software is

(c) Copyright 1992 by Panagiotis Tsirigotis

The author (Panagiotis Tsirigotis) grants permission to use, copy, and distribute this software and its documentation for any purpose and without fee, provided that the above copyright notice extant in files in this distribution is not removed from files included in any redistribution and that this copyright notice is also included in any redistribution.

Modifications to this software may be distributed, either by distributing the modified software or by distributing patches to the original software, under the following additional terms:

1. The version number will be modified as follows:
 - a. The first 3 components of the version number (i.e <number>.<number>.<number>) will remain unchanged.
 - b. A new component will be appended to the version number to indicate the modification level. The form of this component is up to the author of the modifications.
2. The author of the modifications will include his/her name by appending it along with the new version number to this file and will be responsible for any wrong behavior of the modified software.

The author makes no representations about the suitability of this software for any purpose. It is provided "as is" without any express or implied warranty.

Modifications:

Version: 2.1.8.7-current

Copyright 1998-2001 by Rob Braun

Sensor Addition

Version: 2.1.8.9pre14a

Copyright 2001 by Steve Grubb

This is an excerpt from an email I received from the original author, allowing xinetd as maintained by me, to use the higher version numbers:

I appreciate your maintaining the version string guidelines as specified in the copyright. But I did not mean them to last as long as they did.

So, if you want, you may use any 2.N.* (N >= 3) version string for future

xinetd versions that you release. Note that I am excluding the 2.2.* line; using that would only create confusion. Naming the next release 2.3.0 would put to rest the confusion about 2.2.1 and 2.1.8.*.

1.151 xz

1.151.1 Available under license

XZ Utils Licensing

=====

Different licenses apply to different files in this package. Here is a rough summary of which licenses apply to which parts of this package (but check the individual files to be sure!):

- liblzma is in the public domain.
- xz, xzdec, and lzmadec command line tools are in the public domain unless GNU getopt_long had to be compiled and linked in from the lib directory. The getopt_long code is under GNU LGPLv2.1+.
- The scripts to grep, diff, and view compressed files have been adapted from gzip. These scripts and their documentation are under GNU GPLv2+.
- All the documentation in the doc directory and most of the XZ Utils specific documentation files in other directories are in the public domain.
- Translated messages are in the public domain.
- The build system contains public domain files, and files that are under GNU GPLv2+ or GNU GPLv3+. None of these files end up in the binaries being built.
- Test files and test code in the tests directory, and debugging utilities in the debug directory are in the public domain.
- The extra directory may contain public domain files, and files that are under various free software licenses.

You can do whatever you want with the files that have been put into the public domain. If you find public domain legally problematic, take the previous sentence as a license grant. If you still find the lack of copyright legally problematic, you have too many lawyers.

As usual, this software is provided "as is", without any warranty.

If you copy significant amounts of public domain code from XZ Utils into your project, acknowledging this somewhere in your software is polite (especially if it is proprietary, non-free software), but naturally it is not legally required. Here is an example of a good

notice to put into "about box" or into documentation:

This software includes code from XZ Utils <<http://tukaani.org/xz/>>.

The following license texts are included in the following files:

- COPYING.LGPLv2.1: GNU Lesser General Public License version 2.1
- COPYING.GPLv2: GNU General Public License version 2
- COPYING.GPLv3: GNU General Public License version 3

Note that the toolchain (compiler, linker etc.) may add some code pieces that are copyrighted. Thus, it is possible that e.g. liblzma binary wouldn't actually be in the public domain in its entirety even though it contains no copyrighted code from the XZ Utils source package.

If you have questions, don't hesitate to ask the author(s) for more information.

/* Getopt for GNU.

NOTE: getopt is now part of the C library, so if you don't know what "Keep this file name-space clean" means, talk to drepper@gnu.org before changing it!

Copyright (C) 1987,88,89,90,91,92,93,94,95,96,98,99,2000,2001,2002,2003,2004,2006
Free Software Foundation, Inc.

This file is part of the GNU C Library.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA. */

```
#ifndef _LIBC
# include <config.h>
#endif
```

```
#include "getopt.h"
```

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
```

```
#ifdef __VMS
# include <unixlib.h>
#endif
```

```
/* Completely disable NLS for getopt. We won't include translations for it
   anyway. If the system lacks getopt_long, missing translations probably
   aren't a problem. */
```

```
/*
#ifdef _LIBC
#include <libintl.h>
#else
#include "gettext.h"
#define _(msgid) gettext (msgid)
#endif
*/
#define _(msgid) (msgid)
```

```
#if defined _LIBC && defined USE_IN_LIBIO
#include <wchar.h>
#endif
```

```
#ifndef attribute_hidden
#define attribute_hidden
#endif
```

```
/* Unlike standard Unix `getopt', functions like `getopt_long'
   let the user intersperse the options with the other arguments.
```

As `getopt_long' works, it permutes the elements of ARGV so that, when it is done, all the options precede everything else. Thus all application programs are extended to handle flexible argument order.

Using `getopt' or setting the environment variable POSIXLY_CORRECT disables permutation. Then the application's behavior is completely standard.

GNU application programs can use a third alternative mode in which they can distinguish the relative order of options and other arguments. */

```
#include "getopt_int.h"
```

```
/* For communication from `getopt' to the caller.
   When `getopt' finds an option that takes an argument,
   the argument value is returned here.
   Also, when `ordering' is RETURN_IN_ORDER,
   each non-option ARGV-element is returned here. */
```

```
char *optarg;
```

```
/* Index in ARGV of the next element to be scanned.
   This is used for communication to and from the caller
   and for communication between successive calls to `getopt'.
```

On entry to `getopt', zero means this is the first call; initialize.

When `getopt' returns -1, this is the index of the first of the non-option elements that the caller should itself scan.

```

    Otherwise, `optind' communicates from one call to the next
    how much of ARGV has been scanned so far. */

/* 1003.2 says this must be 1 before any call. */
int optind = 1;

/* Callers store zero here to inhibit the error message
   for unrecognized options. */

int opterr = 1;

/* Set to an option character which was unrecognized.
   This must be initialized on some systems to avoid linking in the
   system's own getopt implementation. */

int optopt = '?';

/* Keep a global copy of all internal members of getopt_data. */

static struct __getopt_data getopt_data;

#ifdef HAVE_DECL_GETENV && !HAVE_DECL_GETENV
extern char *getenv ();
#endif

#ifdef _LIBC
/* Stored original parameters.
   XXX This is no good solution. We should rather copy the args so
   that we can compare them later. But we must not use malloc(3). */
extern int __libc_argc;
extern char ** __libc_argv;

/* Bash 2.0 gives us an environment variable containing flags
   indicating ARGV elements that should not be considered arguments. */

#ifdef USE_NONOPTION_FLAGS
/* Defined in getopt_init.c */
extern char *__getopt_nonoption_flags;
#endif

#ifdef USE_NONOPTION_FLAGS
# define SWAP_FLAGS(ch1, ch2) \
    if (d->__nonoption_flags_len > 0) \
    { \
        char __tmp = __getopt_nonoption_flags[ch1]; \
        __getopt_nonoption_flags[ch1] = __getopt_nonoption_flags[ch2]; \
        __getopt_nonoption_flags[ch2] = __tmp; \
    } \
# else
# define SWAP_FLAGS(ch1, ch2)
# endif
#else/* !_LIBC */
# define SWAP_FLAGS(ch1, ch2)
# endif/* !_LIBC */

```



```

/* Exchange two adjacent subsequences of ARGV.
   One subsequence is elements [first_nonopt,last_nonopt)
   which contains all the non-options that have been skipped so far.
   The other is elements [last_nonopt,optind), which contains all
   the options processed since those non-options were skipped.

   `first_nonopt' and `last_nonopt' are relocated so that they describe
   the new indices of the non-options in ARGV after they are moved. */

static void
exchange (char **argv, struct _getopt_data *d)
{
    int bottom = d->__first_nonopt;
    int middle = d->__last_nonopt;
    int top = d->optind;
    char *tem;

    /* Exchange the shorter segment with the far end of the longer segment.
       That puts the shorter segment into the right place.
       It leaves the longer segment in the right place overall,
       but it consists of two parts that need to be swapped next. */

    #if defined _LIBC && defined USE_NONOPTION_FLAGS
    /* First make sure the handling of the `__getopt_nonoption_flags'
       string can work normally. Our top argument must be in the range
       of the string. */
    if (d->__nonoption_flags_len > 0 && top >= d->__nonoption_flags_max_len)
    {
        /* We must extend the array. The user plays games with us and
           presents new arguments. */
        char *new_str = malloc (top + 1);
        if (new_str == NULL)
            d->__nonoption_flags_len = d->__nonoption_flags_max_len = 0;
        else
        {
            memset (__mempcpy (new_str, __getopt_nonoption_flags,
                               d->__nonoption_flags_max_len),
                    '\0', top + 1 - d->__nonoption_flags_max_len);
            d->__nonoption_flags_max_len = top + 1;
            __getopt_nonoption_flags = new_str;
        }
    }
    #endif

    while (top > middle && middle > bottom)
    {
        if (top - middle > middle - bottom)
        {
            /* Bottom segment is the short one. */
            int len = middle - bottom;
            register int i;

            /* Swap it with the top part of the top segment. */

```

```

    for (i = 0; i < len; i++)
    {
        tem = argv[bottom + i];
        argv[bottom + i] = argv[top - (middle - bottom) + i];
        argv[top - (middle - bottom) + i] = tem;
        SWAP_FLAGS (bottom + i, top - (middle - bottom) + i);
    }
    /* Exclude the moved bottom segment from further swapping. */
    top -= len;
}
else
{
    /* Top segment is the short one. */
    int len = top - middle;
    register int i;

    /* Swap it with the bottom part of the bottom segment. */
    for (i = 0; i < len; i++)
    {
        tem = argv[bottom + i];
        argv[bottom + i] = argv[middle + i];
        argv[middle + i] = tem;
        SWAP_FLAGS (bottom + i, middle + i);
    }
    /* Exclude the moved top segment from further swapping. */
    bottom += len;
}

/* Update records for the slots the non-options now occupy. */

d->__first_nopt += (d->optind - d->__last_nopt);
d->__last_nopt = d->optind;
}

/* Initialize the internal data when the first call is made. */

static const char *
_getopt_initialize (int argc, char **argv, const char *optstring,
    int posixly_correct, struct _getopt_data *d)
{
    /* Start processing options with ARGV-element 1 (since ARGV-element 0
       is the program name); the sequence of previously skipped
       non-option ARGV-elements is empty. */

    d->__first_nopt = d->__last_nopt = d->optind;

    d->__nextchar = NULL;

    d->__posixly_correct = posixly_correct || !!getenv ("POSIXLY_CORRECT");

    /* Determine how to handle the ordering of options and nonoptions. */

    if (optstring[0] == '-')
    {

```

```

    d->__ordering = RETURN_IN_ORDER;
    ++optstring;
}
else if (optstring[0] == '+')
{
    d->__ordering = REQUIRE_ORDER;
    ++optstring;
}
else if (d->__posixly_correct)
    d->__ordering = REQUIRE_ORDER;
else
    d->__ordering = PERMUTE;

#if defined _LIBC && defined USE_NONOPTION_FLAGS
if (!d->__posixly_correct
    && argc == __libc_argc && argv == __libc_argv)
{
    if (d->__nonoption_flags_max_len == 0)
    {
        if (__getopt_nonoption_flags == NULL
            || __getopt_nonoption_flags[0] == '\0')
            d->__nonoption_flags_max_len = -1;
        else
        {
            const char *orig_str = __getopt_nonoption_flags;
            int len = d->__nonoption_flags_max_len = strlen (orig_str);
            if (d->__nonoption_flags_max_len < argc)
                d->__nonoption_flags_max_len = argc;
            __getopt_nonoption_flags =
                (char *) malloc (d->__nonoption_flags_max_len);
            if (__getopt_nonoption_flags == NULL)
                d->__nonoption_flags_max_len = -1;
            else
                memset (__mempcpy (__getopt_nonoption_flags, orig_str, len),
                    '\0', d->__nonoption_flags_max_len - len);
        }
    }
    d->__nonoption_flags_len = d->__nonoption_flags_max_len;
}
else
    d->__nonoption_flags_len = 0;
#endif

return optstring;
}

/* Scan elements of ARGV (whose length is ARGV) for option characters
   given in OPTSTRING.

   If an element of ARGV starts with '-', and is not exactly "-" or "--",
   then it is an option element.  The characters of this element
   (aside from the initial '-') are option characters.  If `getopt'
   is called repeatedly, it returns successively each of the option characters
   from each of the option elements.

```

If ``getopt'` finds another option character, it returns that character, updating ``optind'` and ``nextchar'` so that the next call to ``getopt'` can resume the scan with the following option character or ARGV-element.

If there are no more option characters, ``getopt'` returns -1. Then ``optind'` is the index in ARGV of the first ARGV-element that is not an option. (The ARGV-elements have been permuted so that those that are not options now come last.)

OPTSTRING is a string containing the legitimate option characters. If an option character is seen that is not listed in OPTSTRING, return '?' after printing an error message. If you set ``opterr'` to zero, the error message is suppressed but we still return '?'.

If a char in OPTSTRING is followed by a colon, that means it wants an arg, so the following text in the same ARGV-element, or the text of the following ARGV-element, is returned in ``optarg'`. Two colons mean an option that wants an optional arg; if there is text in the current ARGV-element, it is returned in ``optarg'`, otherwise ``optarg'` is set to zero.

If OPTSTRING starts with ``-'` or ``+'`, it requests different methods of handling the non-option ARGV-elements. See the comments about RETURN_IN_ORDER and REQUIRE_ORDER, above.

Long-named options begin with ``--'` instead of ``-'`. Their names may be abbreviated as long as the abbreviation is unique or is an exact match for some defined option. If they have an argument, it follows the option name in the same ARGV-element, separated from the option name by a ``='`, or else the in next ARGV-element. When ``getopt'` finds a long-named option, it returns 0 if that option's ``flag'` field is nonzero, the value of the option's ``val'` field if the ``flag'` field is zero.

LONGOPTS is a vector of ``struct option'` terminated by an element containing a name which is zero.

LONGIND returns the index in LONGOPT of the long-named option found. It is only valid when a long-named option has been found by the most recent call.

If LONG_ONLY is nonzero, ``-'` as well as ``--'` can introduce long-named options.

If POSIXLY_CORRECT is nonzero, behave as if the POSIXLY_CORRECT environment variable were set. **/*

```
int
_getopt_internal_r (int argc, char **argv, const char *optstring,
    const struct option *longopts, int *longind,
    int long_only, int posixly_correct, struct _getopt_data *d)
{
    int print_errors = d->opterr;
    if (optstring[0] == ':')
        print_errors = 0;
```

```

if (argc < 1)
    return -1;

d->optarg = NULL;

if (d->optind == 0 || !d->__initialized)
{
    if (d->optind == 0)
d->optind = 1; /* Don't scan ARGV[0], the program name. */
    optarg = __getopt_initialize (argc, argv, optarg,
        posixly_correct, d);
    d->__initialized = 1;
}

/* Test whether ARGV[optind] points to a non-option argument.
   Either it does not have option syntax, or there is an environment flag
   from the shell indicating it is not an option. The later information
   is only used when the used in the GNU libc. */
#ifdef _LIBC && defined USE_NONOPTION_FLAGS
#define NONOPTION_P (argv[d->optind][0] != '-' || argv[d->optind][1] == '\0' \
    || (d->optind < d->__nonoption_flags_len \
    && __getopt_nonoption_flags[d->optind] == '1'))
#else
#define NONOPTION_P (argv[d->optind][0] != '-' || argv[d->optind][1] == '\0')
#endif

if (d->__nextchar == NULL || *d->__nextchar == '\0')
{
    /* Advance to the next ARGV-element. */

    /* Give FIRST_NONOPT & LAST_NONOPT rational values if OPTIND has been
       moved back by the user (who may also have changed the arguments). */
    if (d->__last_nonopt > d->optind)
d->__last_nonopt = d->optind;
    if (d->__first_nonopt > d->optind)
d->__first_nonopt = d->optind;

    if (d->__ordering == PERMUTE)
    {
        /* If we have just processed some options following some non-options,
           exchange them so that the options come first. */

        if (d->__first_nonopt != d->__last_nonopt
            && d->__last_nonopt != d->optind)
            exchange ((char **) argv, d);
        else if (d->__last_nonopt != d->optind)
            d->__first_nonopt = d->optind;

        /* Skip any additional non-options
           and extend the range of non-options previously skipped. */

        while (d->optind < argc && NONOPTION_P)
            d->optind++;
    }
}

```

```

    d->__last_nonopt = d->optind;
}

/* The special ARGV-element '--' means premature end of options.
Skip it like a null option,
then exchange with previous non-options as if it were an option,
then skip everything else like a non-option. */

    if (d->optind != argc && !strcmp (argv[d->optind], "--"))
    {
        d->optind++;

        if (d->__first_nonopt != d->__last_nonopt
            && d->__last_nonopt != d->optind)
            exchange ((char **) argv, d);
        else if (d->__first_nonopt == d->__last_nonopt)
            d->__first_nonopt = d->optind;
        d->__last_nonopt = argc;

        d->optind = argc;
    }

/* If we have done all the ARGV-elements, stop the scan
and back over any non-options that we skipped and permuted. */

    if (d->optind == argc)
    {
        /* Set the next-arg-index to point at the non-options
        that we previously skipped, so the caller will digest them. */
        if (d->__first_nonopt != d->__last_nonopt)
            d->optind = d->__first_nonopt;
        return -1;
    }

/* If we have come to a non-option and did not permute it,
either stop the scan or describe it to the caller and pass it by. */

    if (NONOPTION_P)
    {
        if (d->__ordering == REQUIRE_ORDER)
            return -1;
        d->optarg = argv[d->optind++];
        return 1;
    }

/* We have found another option-ARGV-element.
Skip the initial punctuation. */

    d->__nextchar = (argv[d->optind] + 1
+ (longopts != NULL && argv[d->optind][1] == '.'));
}

/* Decode the current option-ARGV-element. */

/* Check whether the ARGV-element is a long option.

```

If `long_only` and the ARGV-element has the form `"-f"`, where `f` is a valid short option, don't consider it an abbreviated form of a long option that starts with `f`. Otherwise there would be no way to give the `-f` short option.

On the other hand, if there's a long option `"fubar"` and the ARGV-element is `"-fu"`, do consider that an abbreviation of the long option, just like `--fu`, and not `-f` with arg `"u"`.

This distinction seems to be the most useful approach. */

```

if (longopts != NULL
    && (argv[d->optind][1] == '-')
    || (long_only && (argv[d->optind][2]
        || !strchr (optstring, argv[d->optind][1])))
    {
    char *nameend;
    const struct option *p;
    const struct option *pfound = NULL;
    int exact = 0;
    int ambig = 0;
    int indfound = -1;
    int option_index;

    for (nameend = d->__nextchar; *nameend && *nameend != '='; nameend++)
/* Do nothing. */;

    /* Test all long options for either exact match
or abbreviated matches. */
    for (p = longopts, option_index = 0; p->name; p++, option_index++)
if (!strcmp (p->name, d->__nextchar, nameend - d->__nextchar))
    {
        if ((unsigned int) (nameend - d->__nextchar)
            == (unsigned int) strlen (p->name))
        {
/* Exact match found. */
pfound = p;
indfound = option_index;
exact = 1;
break;
        }
        else if (pfound == NULL)
        {
/* First nonexact match found. */
pfound = p;
indfound = option_index;
        }
        else if (long_only
            || pfound->has_arg != p->has_arg
            || pfound->flag != p->flag
            || pfound->val != p->val)
/* Second or later nonexact match found. */
ambig = 1;
    }
}

```

```

    }

    if (ambig && !exact)
    {
        if (print_errors)
        {
            #if defined _LIBC && defined USE_IN_LIBIO
                char *buf;

                if (__asprintf (&buf, _("%s: option '%s' is ambiguous\n"),
                    argv[0], argv[d->optind]) >= 0)
                {
                    _IO_flockfile (stderr);

                    int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
                    ((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

                    __fxprintf (NULL, "%s", buf);

                    ((_IO_FILE *) stderr)->_flags2 = old_flags2;
                    _IO_funlockfile (stderr);

                    free (buf);
                }
            #else
                fprintf (stderr, _("%s: option '%s' is ambiguous\n"),
                    argv[0], argv[d->optind]);
            #endif
        }
        d->__nextchar += strlen (d->__nextchar);
        d->optind++;
        d->optopt = 0;
        return "?";
    }

    if (pfound != NULL)
    {
        option_index = indfound;
        d->optind++;
        if (*nameend)
        {
            /* Don't test has_arg with >, because some C compilers don't
            allow it to be used on enums. */
            if (pfound->has_arg)
                d->optarg = nameend + 1;
            else
            {
                if (print_errors)
                {
                    #if defined _LIBC && defined USE_IN_LIBIO
                        char *buf;
                        int n;
                    #endif
                }

                if (argv[d->optind - 1][1] == '-')

```



```

{
    /* --option */
    #if defined _LIBC && defined USE_IN_LIBIO
        n = __asprintf (&buf, _("\\
%s: option `--%s' doesn't allow an argument\\n"),
            argv[0], pfound->name);
    #else
        fprintf (stderr, _("\\
%s: option `--%s' doesn't allow an argument\\n"),
            argv[0], pfound->name);
    #endif
}

else
{
    /* +option or -option */
    #if defined _LIBC && defined USE_IN_LIBIO
        n = __asprintf (&buf, _("\\
%s: option `-%c%s' doesn't allow an argument\\n"),
            argv[0], argv[d->optind - 1][0],
            pfound->name);
    #else
        fprintf (stderr, _("\\
%s: option `-%c%s' doesn't allow an argument\\n"),
            argv[0], argv[d->optind - 1][0],
            pfound->name);
    #endif
}

#if defined _LIBC && defined USE_IN_LIBIO
    if (n >= 0)
    {
        _IO_flockfile (stderr);

        int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
        ((_IO_FILE *) stderr)->_flags2
            |= _IO_FLAGS2_NOTCANCEL;

        __fxprintf (NULL, "%s", buf);

        ((_IO_FILE *) stderr)->_flags2 = old_flags2;
        _IO_funlockfile (stderr);

        free (buf);
    }
#endif

    d->__nextchar += strlen (d->__nextchar);

    d->optopt = pfound->val;
    return '?';
}

}

else if (pfound->has_arg == 1)

```

```

    {
        if (d->optind < argc)
            d->optarg = argv[d->optind++];
        else
        {
            if (print_errors)
            {
                #if defined _LIBC && defined USE_IN_LIBIO
                    char *buf;

                    if (__asprintf (&buf, _("\
%s: option '%s' requires an argument\n"),
                        argv[0], argv[d->optind - 1]) >= 0)
                    {
                        _IO_flockfile (stderr);

                        int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
                        ((_IO_FILE *) stderr)->_flags2
                            |= _IO_FLAGS2_NOTCANCEL;

                        __fxprintf (NULL, "%s", buf);

                        ((_IO_FILE *) stderr)->_flags2 = old_flags2;
                        _IO_funlockfile (stderr);

                        free (buf);
                    }
                #else
                    fprintf (stderr,
                        _("\
%s: option '%s' requires an argument\n"),
                        argv[0], argv[d->optind - 1]);
                #endif
            }
            d->__nextchar += strlen (d->__nextchar);
            d->optopt = pfound->val;
            return optstring[0] == ':' ? ':' : '?';
        }
    }
    d->__nextchar += strlen (d->__nextchar);
    if (longind != NULL)
        *longind = option_index;
    if (pfound->flag)
    {
        *(pfound->flag) = pfound->val;
        return 0;
    }
    return pfound->val;
}

/* Can't find it as a long option. If this is not getopt_long_only,
or the option starts with '--' or is not a valid short
option, then it's an error.
Otherwise interpret it as a short option. */
if (!long_only || argv[d->optind][1] == '-')
    || strchr (optstring, *d->__nextchar) == NULL)

```

```

{
    if (print_errors)
    {
#ifdef defined _LIBC && defined USE_IN_LIBIO
        char *buf;
        int n;
#endif

        if (argv[d->optind][1] == '-')
        {
            /* --option */
#ifdef defined _LIBC && defined USE_IN_LIBIO
            n = __asprintf (&buf, _("%s: unrecognized option `--%s'\n"),
                argv[0], d->__nextchar);
            else
            fprintf (stderr, _("%s: unrecognized option `--%s'\n"),
                argv[0], d->__nextchar);
            endif
        }
        else
        {
            /* +option or -option */
#ifdef defined _LIBC && defined USE_IN_LIBIO
            n = __asprintf (&buf, _("%s: unrecognized option `%c%s'\n"),
                argv[0], argv[d->optind][0], d->__nextchar);
            else
            fprintf (stderr, _("%s: unrecognized option `%c%s'\n"),
                argv[0], argv[d->optind][0], d->__nextchar);
            endif
        }

#ifdef defined _LIBC && defined USE_IN_LIBIO
        if (n >= 0)
        {
            _IO_flockfile (stderr);

            int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
            ((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

            __fxprintf (NULL, "%s", buf);

            ((_IO_FILE *) stderr)->_flags2 = old_flags2;
            _IO_funlockfile (stderr);

            free (buf);
        }
#endif
    }
    d->__nextchar = (char *) "";
    d->optind++;
    d->optopt = 0;
    return '?';
}
}

```

```

/* Look at and handle the next short option-character. */

{
    char c = *d->__nextchar++;
    char *temp = strchr (optstring, c);

    /* Increment 'optind' when we start to process its last character. */
    if (*d->__nextchar == '\0')
        ++d->optind;

    if (temp == NULL || c == ':')
    {
        if (print_errors)
        {
#ifdef defined _LIBC && defined USE_IN_LIBIO
            char *buf;
            int n;
#endif

            if (d->__posixly_correct)
            {
                /* 1003.2 specifies the format of this message. */
#ifdef defined _LIBC && defined USE_IN_LIBIO
                n = __asprintf (&buf, _("%s: illegal option -- %c\n"),
                                argv[0], c);
                argv[0], c);
                #else
                fprintf (stderr, _("%s: illegal option -- %c\n"), argv[0], c);
                #endif
            }
            else
            {
#ifdef defined _LIBC && defined USE_IN_LIBIO
                n = __asprintf (&buf, _("%s: invalid option -- %c\n"),
                                argv[0], c);
                argv[0], c);
                #else
                fprintf (stderr, _("%s: invalid option -- %c\n"), argv[0], c);
                #endif
            }
        }

#ifdef defined _LIBC && defined USE_IN_LIBIO
        if (n >= 0)
        {
            _IO_flockfile (stderr);

            int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
            ((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

            __fxprintf (NULL, "%s", buf);

            ((_IO_FILE *) stderr)->_flags2 = old_flags2;
            _IO_funlockfile (stderr);

            free (buf);
        }

```

```

#endif
}
d->optopt = c;
return '?';
}
/* Convenience. Treat POSIX -W foo same as long option --foo */
if (temp[0] == 'W' && temp[1] == ';')
{
char *nameend;
const struct option *p;
const struct option *pfound = NULL;
int exact = 0;
int ambig = 0;
int indfound = 0;
int option_index;

/* This is an option that requires an argument. */
if (*d->__nextchar != '\0')
{
d->optarg = d->__nextchar;
/* If we end this ARGV-element by taking the rest as an arg,
we must advance to the next element now. */
d->optind++;
}
else if (d->optind == argc)
{
if (print_errors)
{
/* 1003.2 specifies the format of this message. */
#if defined _LIBC && defined USE_IN_LIBIO
char *buf;

if (__asprintf (&buf,
_("%s: option requires an argument -- %c\n"),
argv[0], c) >= 0)
{
_IO_flockfile (stderr);

int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

__fxprintf (NULL, "%s", buf);

((_IO_FILE *) stderr)->_flags2 = old_flags2;
_IO_funlockfile (stderr);

free (buf);
}
}
#else
fprintf (stderr, _("%s: option requires an argument -- %c\n"),
argv[0], c);
#endif
}
}
d->optopt = c;

```

```

        if (optstring[0] == ':')
            c = ':';
        else
            c = '?';
        return c;
    }
else
    /* We already incremented 'd->optind' once;
       increment it again when taking next ARGV-elt as argument. */
    d->optarg = argv[d->optind++];

/* optarg is now the argument, see if it's in the
   table of longopts. */

for (d->__nextchar = nameend = d->optarg; *nameend && *nameend != '=';
     nameend++)
    /* Do nothing. */;

/* Test all long options for either exact match
   or abbreviated matches. */
for (p = longopts, option_index = 0; p->name; p++, option_index++)
    if (!strcmp (p->name, d->__nextchar, nameend - d->__nextchar))
    {
        if ((unsigned int) (nameend - d->__nextchar) == strlen (p->name))
        {
            /* Exact match found. */
            pfound = p;
            indfound = option_index;
            exact = 1;
            break;
        }
        else if (pfound == NULL)
        {
            /* First nonexact match found. */
            pfound = p;
            indfound = option_index;
        }
        else
            /* Second or later nonexact match found. */
            ambig = 1;
    }
if (ambig && !exact)
{
    if (print_errors)
    {
#ifdef _LIBC && defined USE_IN_LIBIO
        char *buf;

        if (__asprintf (&buf, _("%s: option '-W %s' is ambiguous\n"),
            argv[0], argv[d->optind]) >= 0)
        {
            _IO_flockfile (stderr);

            int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
            ((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

```

```

    __fxprintf (NULL, "%s", buf);

    ((_IO_FILE *) stderr)->_flags2 = old_flags2;
    _IO_funlockfile (stderr);

    free (buf);
}
#else
fprintf (stderr, _("%s: option `-W %s' is ambiguous\n"),
        argv[0], argv[d->optind]);
#endif
}
d->__nextchar += strlen (d->__nextchar);
d->optind++;
return '?';
}
if (pfound != NULL)
{
    option_index = indfound;
    if (*nameend)
    {
        /* Don't test has_arg with >, because some C compilers don't
        allow it to be used on enums. */
        if (pfound->has_arg)
            d->optarg = nameend + 1;
        else
        {
            if (print_errors)
            {
#ifdef defined _LIBC && defined USE_IN_LIBIO
char *buf;

if (__asprintf (&buf, _("\
%s: option `-W %s' doesn't allow an argument\n"),
        argv[0], pfound->name) >= 0)
{
    _IO_flockfile (stderr);

    int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
    ((_IO_FILE *) stderr)->_flags2
        |= _IO_FLAGS2_NOTCANCEL;

    __fxprintf (NULL, "%s", buf);

    ((_IO_FILE *) stderr)->_flags2 = old_flags2;
    _IO_funlockfile (stderr);

    free (buf);
}
#endif
            if (print_errors)
            {
                if (pfound->name[0] == '-')
                    fprintf (stderr, _("%s: option `%s' is ambiguous\n"),
                            argv[0], pfound->name);
                else
                    fprintf (stderr, _("%s: option `%s' doesn't allow an argument\n"),
                            argv[0], pfound->name);
            }
        }
    }
}

```

```

#endif
}

d->__nextchar += strlen (d->__nextchar);
return '?';
}
}
else if (pfound->has_arg == 1)
{
if (d->optind < argc)
d->optarg = argv[d->optind++];
else
{
if (print_errors)
{
#if defined _LIBC && defined USE_IN_LIBIO
char *buf;

if (__asprintf (&buf, _("
%s: option '%s' requires an argument\n"),
argv[0], argv[d->optind - 1]) >= 0)
{
_IO_flockfile (stderr);

int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
((_IO_FILE *) stderr)->_flags2
|= _IO_FLAGS2_NOTCANCEL;

__fxprintf (NULL, "%s", buf);

((_IO_FILE *) stderr)->_flags2 = old_flags2;
_IO_funlockfile (stderr);

free (buf);
}
}
#else
fprintf (stderr,
_("
%s: option '%s' requires an argument\n"),
argv[0], argv[d->optind - 1]);
#endif
}
d->__nextchar += strlen (d->__nextchar);
return optstring[0] == ':' ? ':' : '?';
}
}
d->__nextchar += strlen (d->__nextchar);
if (longind != NULL)
*longind = option_index;
if (pfound->flag)
{
*(pfound->flag) = pfound->val;
return 0;
}
return pfound->val;
}

```



```

d->__nextchar = NULL;
return 'W';/* Let the application handle it. */
}
if (temp[1] == ':')
{
if (temp[2] == ':')
{
/* This is an option that accepts an argument optionally. */
if (*d->__nextchar != '\0')
{
d->optarg = d->__nextchar;
d->optind++;
}
else
d->optarg = NULL;
d->__nextchar = NULL;
}
else
{
/* This is an option that requires an argument. */
if (*d->__nextchar != '\0')
{
d->optarg = d->__nextchar;
/* If we end this ARGV-element by taking the rest as an arg,
we must advance to the next element now. */
d->optind++;
}
else if (d->optind == argc)
{
if (print_errors)
{
/* 1003.2 specifies the format of this message. */
#ifdef _LIBC && defined USE_IN_LIBIO
char *buf;

if (__asprintf (&buf, _("\\
%s: option requires an argument -- %c\\n"),
argv[0], c) >= 0)
{
_IO_flockfile (stderr);

int old_flags2 = ((_IO_FILE *) stderr)->_flags2;
((_IO_FILE *) stderr)->_flags2 |= _IO_FLAGS2_NOTCANCEL;

__fxprintf (NULL, "%s", buf);

((_IO_FILE *) stderr)->_flags2 = old_flags2;
_IO_funlockfile (stderr);

free (buf);
}
}
else
fprintf (stderr,
_("\\
%s: option requires an argument -- %c\\n"),

```

```

    argv[0], c);
#endif
}
d->optopt = c;
if (optstring[0] == ':')
    c = ':';
else
    c = '?';
}
else
    /* We already incremented `optind' once;
    increment it again when taking next ARGV-elt as argument. */
    d->optarg = argv[d->optind++];
    d->__nextchar = NULL;
}
}
return c;
}
}

int
_getopt_internal (int argc, char **argv, const char *optstring,
const struct option *longopts, int *longind,
int long_only, int posixly_correct)
{
    int result;

    getopt_data.optind = optind;
    getopt_data.opterr = opterr;

    result = _getopt_internal_r (argc, argv, optstring, longopts, longind,
        long_only, posixly_correct, &getopt_data);

    optind = getopt_data.optind;
    optarg = getopt_data.optarg;
    optopt = getopt_data.optopt;

    return result;
}

/* glibc gets a LSB-compliant getopt.
   Standalone applications get a POSIX-compliant getopt. */
#if _LIBC
enum { POSIXLY_CORRECT = 0 };
#else
enum { POSIXLY_CORRECT = 1 };
#endif

int
getopt (int argc, char *const *argv, const char *optstring)
{
    return _getopt_internal (argc, (char **) argv, optstring, NULL, NULL, 0,
        POSIXLY_CORRECT);
}

```

```
#ifdef TEST

/* Compile with -DTEST to make an executable for use in testing
   the above definition of `getopt'. */

int
main (int argc, char **argv)
{
  int c;
  int digit_optind = 0;

  while (1)
    {
      int this_option_optind = optind ? optind : 1;

      c = getopt (argc, argv, "abc:d:0123456789");
      if (c == -1)
        break;

      switch (c)
        {
        case '0':
        case '1':
        case '2':
        case '3':
        case '4':
        case '5':
        case '6':
        case '7':
        case '8':
        case '9':
          if (digit_optind != 0 && digit_optind != this_option_optind)
            printf ("digits occur in two different argv-elements.\n");
          digit_optind = this_option_optind;
          printf ("option %c\n", c);
          break;

        case 'a':
          printf ("option a\n");
          break;

        case 'b':
          printf ("option b\n");
          break;

        case 'c':
          printf ("option c with value `%s'\n", optarg);
          break;

        case '?':
          break;

        default:

```

```
    printf ("?? getopt returned character code 0%o ??\n", c);
}
}

if (optind < argc)
{
    printf ("non-option ARGV-elements: ");
    while (optind < argc)
        printf ("%s ", argv[optind++]);
    printf ("\n");
}

exit (0);
}

#endif /* TEST */
```

1.152 zip

1.152.1 Available under license

This is version 2007-Mar-4 of the Info-ZIP license.

The definitive version of this document should be available at <ftp://ftp.info-zip.org/pub/infozip/license.html> indefinitely and a copy at <http://www.info-zip.org/pub/infozip/license.html>.

Copyright (c) 1990-2007 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda, Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the above disclaimer and the following restrictions:

1. Redistributions of source code (in whole or in part) must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2. Redistributions in binary form (compiled executables and libraries) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.
3. Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, versions with modified or added functionality, and dynamic, shared, or static library versions not from Info-ZIP--must be plainly marked as such and must not be misrepresented as being the original source or, if binaries,

compiled from the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but not limited to, labeling of the altered versions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP will provide support for the altered versions.

4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

1.153 zlib

1.153.1 Available under license

/* zlib.h -- interface of the 'zlib' general purpose compression library
version 1.2.8, April 28th, 2013

Copyright (C) 1995-2013 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly	Mark Adler
jloup@gzip.org	madler@alumni.caltech.edu

The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files <http://tools.ietf.org/html/rfc1950> (zlib format), rfc1951 (deflate format) and rfc1952 (gzip format).

*/

```
#ifndef ZLIB_H
#define ZLIB_H
```

```
#include "zconf.h"
```

```
#ifdef __cplusplus
extern "C" {
#endif
```

```
#define ZLIB_VERSION "1.2.8"
#define ZLIB_VERNUM 0x1280
#define ZLIB_VER_MAJOR 1
#define ZLIB_VER_MINOR 2
#define ZLIB_VER_REVISION 8
#define ZLIB_VER_SUBREVISION 0
```

/*

The 'zlib' compression library provides in-memory compression and decompression functions, including integrity checks of the uncompressed data. This version of the library supports only one compression method (deflation) but other algorithms will be added later and will have the same stream interface.

Compression can be done in a single step if the buffers are large enough, or can be done by repeated calls of the compression function. In the latter case, the application must provide more input and/or consume the output (providing more output space) before each call.

The compressed data format used by default by the in-memory functions is the zlib format, which is a zlib wrapper documented in RFC 1950, wrapped around a deflate stream, which is itself documented in RFC 1951.

The library also supports reading and writing files in gzip (.gz) format with an interface similar to that of stdio using the functions that start with "gz". The gzip format is different from the zlib format. gzip is a gzip wrapper, documented in RFC 1952, wrapped around a deflate stream.

This library can optionally read and write gzip streams in memory as well.

The zlib format was designed to be compact and fast for use in memory and on communications channels. The gzip format was designed for single-file compression on file systems, has a larger header than zlib to maintain directory information, and uses a different, slower check method than zlib.

The library does not install any signal handler. The decoder checks the consistency of the compressed data, so the library should never crash even in case of corrupted input.

*/

```
typedef voidpf (*alloc_func) OF((voidpf opaque, uInt items, uInt size));
```

```
typedef void (*free_func) OF((voidpf opaque, voidpf address));
```

```
struct internal_state;
```

```
typedef struct z_stream_s {
```

```
    z_const Bytef *next_in; /* next input byte */
```

```
    uInt avail_in; /* number of bytes available at next_in */
```

```
    uLong total_in; /* total number of input bytes read so far */
```

```
    Bytef *next_out; /* next output byte should be put there */
```

```
    uInt avail_out; /* remaining free space at next_out */
```

```
    uLong total_out; /* total number of bytes output so far */
```

```
    z_const char *msg; /* last error message, NULL if no error */
```

```
    struct internal_state FAR *state; /* not visible by applications */
```

```
    alloc_func zalloc; /* used to allocate the internal state */
```

```
    free_func zfree; /* used to free the internal state */
```

```
    voidpf opaque; /* private data object passed to zalloc and zfree */
```

```
    int data_type; /* best guess about the data type: binary or text */
```

```
    uLong adler; /* Adler-32 value of the uncompressed data */
```



```

    uLong reserved; /* reserved for future use */
} z_stream;

typedef z_stream FAR *z_streamp;

/*
    gzip header information passed to and from zlib routines. See RFC 1952
    for more details on the meanings of these fields.
*/
typedef struct gz_header_s {
    int text; /* true if compressed data believed to be text */
    uLong time; /* modification time */
    int xflags; /* extra flags (not used when writing a gzip file) */
    int os; /* operating system */
    Bytef *extra; /* pointer to extra field or Z_NULL if none */
    uInt extra_len; /* extra field length (valid if extra != Z_NULL) */
    uInt extra_max; /* space at extra (only when reading header) */
    Bytef *name; /* pointer to zero-terminated file name or Z_NULL */
    uInt name_max; /* space at name (only when reading header) */
    Bytef *comment; /* pointer to zero-terminated comment or Z_NULL */
    uInt comm_max; /* space at comment (only when reading header) */
    int hcrc; /* true if there was or will be a header crc */
    int done; /* true when done reading gzip header (not used
               when writing a gzip file) */
} gz_header;

typedef gz_header FAR *gz_headerp;

/*
    The application must update next_in and avail_in when avail_in has dropped
    to zero. It must update next_out and avail_out when avail_out has dropped
    to zero. The application must initialize zalloc, zfree and opaque before
    calling the init function. All other fields are set by the compression
    library and must not be updated by the application.

    The opaque value provided by the application will be passed as the first
    parameter for calls of zalloc and zfree. This can be useful for custom
    memory management. The compression library attaches no meaning to the
    opaque value.

    zalloc must return Z_NULL if there is not enough memory for the object.
    If zlib is used in a multi-threaded application, zalloc and zfree must be
    thread safe.

    On 16-bit systems, the functions zalloc and zfree must be able to allocate
    exactly 65536 bytes, but will not be required to allocate more than this if
    the symbol MAXSEG_64K is defined (see zconf.h). WARNING: On MSDOS, pointers
    returned by zalloc for objects of exactly 65536 bytes *must* have their
    offset normalized to zero. The default allocation function provided by this
    library ensures this (see zutil.c). To reduce memory requirements and avoid
    any allocation of 64K objects, at the expense of compression ratio, compile
    the library with -DMAX_WBITS=14 (see zconf.h).

    The fields total_in and total_out can be used for statistics or progress

```

reports. After compression, total_in holds the total size of the uncompressed data and may be saved for use in the decompressor (particularly if the decompressor wants to decompress everything in a single step).

```

*/

        /* constants */

#define Z_NO_FLUSH      0
#define Z_PARTIAL_FLUSH 1
#define Z_SYNC_FLUSH    2
#define Z_FULL_FLUSH    3
#define Z_FINISH        4
#define Z_BLOCK         5
#define Z_TREES         6
/* Allowed flush values; see deflate() and inflate() below for details */

#define Z_OK             0
#define Z_STREAM_END     1
#define Z_NEED_DICT      2
#define Z_ERRNO          (-1)
#define Z_STREAM_ERROR   (-2)
#define Z_DATA_ERROR     (-3)
#define Z_MEM_ERROR      (-4)
#define Z_BUF_ERROR      (-5)
#define Z_VERSION_ERROR  (-6)
/* Return codes for the compression/decompression functions. Negative values
 * are errors, positive values are used for special but normal events.
 */

#define Z_NO_COMPRESSION      0
#define Z_BEST_SPEED          1
#define Z_BEST_COMPRESSION    9
#define Z_DEFAULT_COMPRESSION (-1)
/* compression levels */

#define Z_FILTERED            1
#define Z_HUFFMAN_ONLY        2
#define Z_RLE                 3
#define Z_FIXED               4
#define Z_DEFAULT_STRATEGY    0
/* compression strategy; see deflateInit2() below for details */

#define Z_BINARY              0
#define Z_TEXT                1
#define Z_ASCII               Z_TEXT /* for compatibility with 1.2.2 and earlier */
#define Z_UNKNOWN             2
/* Possible values of the data_type field (though see inflate()) */

#define Z_DEFLATED            8
/* The deflate compression method (the only one supported in this version) */

#define Z_NULL 0 /* for initializing zalloc, zfree, opaque */

#define zlib_version zlibVersion()
/* for compatibility with versions < 1.0.2 */

```

/* basic functions */

ZEXTERN const char * ZEXPORT zlibVersion OF((void));
 /* The application can compare zlibVersion and ZLIB_VERSION for consistency.
 If the first character differs, the library code actually used is not
 compatible with the zlib.h header file used by the application. This check
 is automatically made by deflateInit and inflateInit.
 */

/*
 ZEXTERN int ZEXPORT deflateInit OF((z_streamp strm, int level));

Initializes the internal stream state for compression. The fields
 zalloc, zfree and opaque must be initialized before by the caller. If
 zalloc and zfree are set to Z_NULL, deflateInit updates them to use default
 allocation functions.

The compression level must be Z_DEFAULT_COMPRESSION, or between 0 and 9:
 1 gives best speed, 9 gives best compression, 0 gives no compression at all
 (the input data is simply copied a block at a time). Z_DEFAULT_COMPRESSION
 requests a default compromise between speed and compression (currently
 equivalent to level 6).

deflateInit returns Z_OK if success, Z_MEM_ERROR if there was not enough
 memory, Z_STREAM_ERROR if level is not a valid compression level, or
 Z_VERSION_ERROR if the zlib library version (zlib_version) is incompatible
 with the version assumed by the caller (ZLIB_VERSION). msg is set to null
 if there is no error message. deflateInit does not perform any compression:
 this will be done by deflate().
 */

ZEXTERN int ZEXPORT deflate OF((z_streamp strm, int flush));
 /*

deflate compresses as much data as possible, and stops when the input
 buffer becomes empty or the output buffer becomes full. It may introduce
 some output latency (reading input without producing any output) except when
 forced to flush.

The detailed semantics are as follows. deflate performs one or both of the
 following actions:

- Compress more input starting at next_in and update next_in and avail_in
 accordingly. If not all input can be processed (because there is not
 enough room in the output buffer), next_in and avail_in are updated and
 processing will resume at this point for the next call of deflate().
- Provide more output starting at next_out and update next_out and avail_out
 accordingly. This action is forced if the parameter flush is non zero.
 Forcing flush frequently degrades the compression ratio, so this parameter
 should be set only when necessary (in interactive applications). Some
 output may be provided even if flush is not set.

Before the call of `deflate()`, the application should ensure that at least one of the actions is possible, by providing more input and/or consuming more output, and updating `avail_in` or `avail_out` accordingly; `avail_out` should never be zero before the call. The application can consume the compressed output when it wants, for example when the output buffer is full (`avail_out == 0`), or after each call of `deflate()`. If `deflate` returns `Z_OK` and with zero `avail_out`, it must be called again after making room in the output buffer because there might be more output pending.

Normally the parameter `flush` is set to `Z_NO_FLUSH`, which allows `deflate` to decide how much data to accumulate before producing output, in order to maximize compression.

If the parameter `flush` is set to `Z_SYNC_FLUSH`, all pending output is flushed to the output buffer and the output is aligned on a byte boundary, so that the decompressor can get all input data available so far. (In particular `avail_in` is zero after the call if enough output space has been provided before the call.) Flushing may degrade compression for some compression algorithms and so it should be used only when necessary. This completes the current `deflate` block and follows it with an empty stored block that is three bits plus filler bits to the next byte, followed by four bytes (00 00 ff ff).

If `flush` is set to `Z_PARTIAL_FLUSH`, all pending output is flushed to the output buffer, but the output is not aligned to a byte boundary. All of the input data so far will be available to the decompressor, as for `Z_SYNC_FLUSH`. This completes the current `deflate` block and follows it with an empty fixed codes block that is 10 bits long. This assures that enough bytes are output in order for the decompressor to finish the block before the empty fixed code block.

If `flush` is set to `Z_BLOCK`, a `deflate` block is completed and emitted, as for `Z_SYNC_FLUSH`, but the output is not aligned on a byte boundary, and up to seven bits of the current block are held to be written as the next byte after the next `deflate` block is completed. In this case, the decompressor may not be provided enough bits at this point in order to complete decompression of the data provided so far to the compressor. It may need to wait for the next block to be emitted. This is for advanced applications that need to control the emission of `deflate` blocks.

If `flush` is set to `Z_FULL_FLUSH`, all output is flushed as with `Z_SYNC_FLUSH`, and the compression state is reset so that decompression can restart from this point if previous compressed data has been damaged or if random access is desired. Using `Z_FULL_FLUSH` too often can seriously degrade compression.

If `deflate` returns with `avail_out == 0`, this function must be called again with the same value of the `flush` parameter and more output space (updated `avail_out`), until the flush is complete (`deflate` returns with non-zero `avail_out`). In the case of a `Z_FULL_FLUSH` or `Z_SYNC_FLUSH`, make sure that `avail_out` is greater than six to avoid repeated flush markers due to `avail_out == 0` on return.

If the parameter `flush` is set to `Z_FINISH`, pending input is processed,

pending output is flushed and deflate returns with Z_STREAM_END if there was enough output space; if deflate returns with Z_OK, this function must be called again with Z_FINISH and more output space (updated avail_out) but no more input data, until it returns with Z_STREAM_END or an error. After deflate has returned Z_STREAM_END, the only possible operations on the stream are deflateReset or deflateEnd.

Z_FINISH can be used immediately after deflateInit if all the compression is to be done in a single step. In this case, avail_out must be at least the value returned by deflateBound (see below). Then deflate is guaranteed to return Z_STREAM_END. If not enough output space is provided, deflate will not return Z_STREAM_END, and it must be called again as described above.

deflate() sets strm->adler to the adler32 checksum of all input read so far (that is, total_in bytes).

deflate() may update strm->data_type if it can make a good guess about the input data type (Z_BINARY or Z_TEXT). In doubt, the data is considered binary. This field is only for information purposes and does not affect the compression algorithm in any manner.

deflate() returns Z_OK if some progress has been made (more input processed or more output produced), Z_STREAM_END if all input has been consumed and all output has been produced (only when flush is set to Z_FINISH), Z_STREAM_ERROR if the stream state was inconsistent (for example if next_in or next_out was Z_NULL), Z_BUF_ERROR if no progress is possible (for example avail_in or avail_out was zero). Note that Z_BUF_ERROR is not fatal, and deflate() can be called again with more input and more output space to continue compressing.

*/

ZEXTERN int ZEXPORT deflateEnd OF((z_streamp strm));

/*

All dynamically allocated data structures for this stream are freed. This function discards any unprocessed input and does not flush any pending output.

deflateEnd returns Z_OK if success, Z_STREAM_ERROR if the stream state was inconsistent, Z_DATA_ERROR if the stream was freed prematurely (some input or output was discarded). In the error case, msg may be set but then points to a static string (which must not be deallocated).

*/

/*

ZEXTERN int ZEXPORT inflateInit OF((z_streamp strm));

Initializes the internal stream state for decompression. The fields next_in, avail_in, zalloc, zfree and opaque must be initialized before by the caller. If next_in is not Z_NULL and avail_in is large enough (the exact value depends on the compression method), inflateInit determines the compression method from the zlib header and allocates all data structures

accordingly; otherwise the allocation will be deferred to the first call of `inflate`. If `zalloc` and `zfree` are set to `Z_NULL`, `inflateInit` updates them to use default allocation functions.

`inflateInit` returns `Z_OK` if success, `Z_MEM_ERROR` if there was not enough memory, `Z_VERSION_ERROR` if the zlib library version is incompatible with the version assumed by the caller, or `Z_STREAM_ERROR` if the parameters are invalid, such as a null pointer to the structure. `msg` is set to null if there is no error message. `inflateInit` does not perform any decompression apart from possibly reading the zlib header if present: actual decompression will be done by `inflate()`. (So `next_in` and `avail_in` may be modified, but `next_out` and `avail_out` are unused and unchanged.) The current implementation of `inflateInit()` does not process any header information -- that is deferred until `inflate()` is called.

*/

ZEXTERN int ZEXPORT inflate OF((z_stream strm, int flush));

/*

`inflate` decompresses as much data as possible, and stops when the input buffer becomes empty or the output buffer becomes full. It may introduce some output latency (reading input without producing any output) except when forced to flush.

The detailed semantics are as follows. `inflate` performs one or both of the following actions:

- Decompress more input starting at `next_in` and update `next_in` and `avail_in` accordingly. If not all input can be processed (because there is not enough room in the output buffer), `next_in` is updated and processing will resume at this point for the next call of `inflate()`.
- Provide more output starting at `next_out` and update `next_out` and `avail_out` accordingly. `inflate()` provides as much output as possible, until there is no more input data or no more space in the output buffer (see below about the flush parameter).

Before the call of `inflate()`, the application should ensure that at least one of the actions is possible, by providing more input and/or consuming more output, and updating the `next_*` and `avail_*` values accordingly. The application can consume the uncompressed output when it wants, for example when the output buffer is full (`avail_out == 0`), or after each call of `inflate()`. If `inflate` returns `Z_OK` and with zero `avail_out`, it must be called again after making room in the output buffer because there might be more output pending.

The flush parameter of `inflate()` can be `Z_NO_FLUSH`, `Z_SYNC_FLUSH`, `Z_FINISH`, `Z_BLOCK`, or `Z_TREES`. `Z_SYNC_FLUSH` requests that `inflate()` flush as much output as possible to the output buffer. `Z_BLOCK` requests that `inflate()` stop if and when it gets to the next deflate block boundary. When decoding the zlib or gzip format, this will cause `inflate()` to return immediately after the header and before the first block. When doing a raw inflate, `inflate()` will go ahead and process the first block, and will return when it gets to the end of that block, or when it runs out of data.

The `Z_BLOCK` option assists in appending to or combining deflate streams. Also to assist in this, on return `inflate()` will set `strm->data_type` to the number of unused bits in the last byte taken from `strm->next_in`, plus 64 if `inflate()` is currently decoding the last block in the deflate stream, plus 128 if `inflate()` returned immediately after decoding an end-of-block code or decoding the complete header up to just before the first byte of the deflate stream. The end-of-block will not be indicated until all of the uncompressed data from that block has been written to `strm->next_out`. The number of unused bits may in general be greater than seven, except when bit 7 of `data_type` is set, in which case the number of unused bits will be less than eight. `data_type` is set as noted here every time `inflate()` returns for all flush options, and so can be used to determine the amount of currently consumed input in bits.

The `Z_TREES` option behaves as `Z_BLOCK` does, but it also returns when the end of each deflate block header is reached, before any actual data in that block is decoded. This allows the caller to determine the length of the deflate block header for later use in random access within a deflate block. 256 is added to the value of `strm->data_type` when `inflate()` returns immediately after reaching the end of the deflate block header.

`inflate()` should normally be called until it returns `Z_STREAM_END` or an error. However if all decompression is to be performed in a single step (a single call of `inflate`), the parameter `flush` should be set to `Z_FINISH`. In this case all pending input is processed and all pending output is flushed; `avail_out` must be large enough to hold all of the uncompressed data for the operation to complete. (The size of the uncompressed data may have been saved by the compressor for this purpose.) The use of `Z_FINISH` is not required to perform an inflation in one step. However it may be used to inform `inflate` that a faster approach can be used for the single `inflate()` call. `Z_FINISH` also informs `inflate` to not maintain a sliding window if the stream completes, which reduces `inflate`'s memory footprint. If the stream does not complete, either because not all of the stream is provided or not enough output space is provided, then a sliding window will be allocated and `inflate()` can be called again to continue the operation as if `Z_NO_FLUSH` had been used.

In this implementation, `inflate()` always flushes as much output as possible to the output buffer, and always uses the faster approach on the first call. So the effects of the flush parameter in this implementation are on the return value of `inflate()` as noted below, when `inflate()` returns early when `Z_BLOCK` or `Z_TREES` is used, and when `inflate()` avoids the allocation of memory for a sliding window when `Z_FINISH` is used.

If a preset dictionary is needed after this call (see `inflateSetDictionary` below), `inflate` sets `strm->adler` to the Adler-32 checksum of the dictionary chosen by the compressor and returns `Z_NEED_DICT`; otherwise it sets `strm->adler` to the Adler-32 checksum of all output produced so far (that is, `total_out` bytes) and returns `Z_OK`, `Z_STREAM_END` or an error code as described below. At the end of the stream, `inflate()` checks that its computed `adler32` checksum is equal to that saved by the compressor and returns `Z_STREAM_END` only if the checksum is correct.

`inflate()` can decompress and check either `zlib`-wrapped or `gzip`-wrapped

deflate data. The header type is detected automatically, if requested when initializing with `inflateInit2()`. Any information contained in the gzip header is not retained, so applications that need that information should instead use raw inflate, see `inflateInit2()` below, or `inflateBack()` and perform their own processing of the gzip header and trailer. When processing gzip-wrapped deflate data, `strm->adler32` is set to the CRC-32 of the output produced so far. The CRC-32 is checked against the gzip trailer.

`inflate()` returns `Z_OK` if some progress has been made (more input processed or more output produced), `Z_STREAM_END` if the end of the compressed data has been reached and all uncompressed output has been produced, `Z_NEED_DICT` if a preset dictionary is needed at this point, `Z_DATA_ERROR` if the input data was corrupted (input stream not conforming to the zlib format or incorrect check value), `Z_STREAM_ERROR` if the stream structure was inconsistent (for example `next_in` or `next_out` was `Z_NULL`), `Z_MEM_ERROR` if there was not enough memory, `Z_BUF_ERROR` if no progress is possible or if there was not enough room in the output buffer when `Z_FINISH` is used. Note that `Z_BUF_ERROR` is not fatal, and `inflate()` can be called again with more input and more output space to continue decompressing. If `Z_DATA_ERROR` is returned, the application may then call `inflateSync()` to look for a good compression block if a partial recovery of the data is desired.

*/

```
ZEXTERN int ZEXPORT inflateEnd OF((z_streamp strm));
```

/*

All dynamically allocated data structures for this stream are freed. This function discards any unprocessed input and does not flush any pending output.

`inflateEnd` returns `Z_OK` if success, `Z_STREAM_ERROR` if the stream state was inconsistent. In the error case, `msg` may be set but then points to a static string (which must not be deallocated).

*/

/* Advanced functions */

/*

The following functions are needed only in some special applications.

*/

/*

```
ZEXTERN int ZEXPORT deflateInit2 OF((z_streamp strm,
                                     int level,
                                     int method,
                                     int windowBits,
                                     int memLevel,
                                     int strategy));
```

This is another version of `deflateInit` with more compression options. The fields `next_in`, `zalloc`, `zfree` and `opaque` must be initialized before by the caller.

The method parameter is the compression method. It must be `Z_DEFLATED` in

this version of the library.

The windowBits parameter is the base two logarithm of the window size (the size of the history buffer). It should be in the range 8..15 for this version of the library. Larger values of this parameter result in better compression at the expense of memory usage. The default value is 15 if deflateInit is used instead.

windowBits can also be -8..-15 for raw deflate. In this case, -windowBits determines the window size. deflate() will then generate raw deflate data with no zlib header or trailer, and will not compute an Adler32 check value.

windowBits can also be greater than 15 for optional gzip encoding. Add 16 to windowBits to write a simple gzip header and trailer around the compressed data instead of a zlib wrapper. The gzip header will have no file name, no extra data, no comment, no modification time (set to zero), no header CRC, and the operating system will be set to 255 (unknown). If a gzip stream is being written, strm->adler is a CRC32 instead of an Adler32.

The memLevel parameter specifies how much memory should be allocated for the internal compression state. memLevel=1 uses minimum memory but is slow and reduces compression ratio; memLevel=9 uses maximum memory for optimal speed. The default value is 8. See zconf.h for total memory usage as a function of windowBits and memLevel.

The strategy parameter is used to tune the compression algorithm. Use the value Z_DEFAULT_STRATEGY for normal data, Z_FILTERED for data produced by a filter (or predictor), Z_HUFFMAN_ONLY to force Huffman encoding only (no string match), or Z_RLE to limit match distances to one (run-length encoding). Filtered data consists mostly of small values with a somewhat random distribution. In this case, the compression algorithm is tuned to compress them better. The effect of Z_FILTERED is to force more Huffman coding and less string matching; it is somewhat intermediate between Z_DEFAULT_STRATEGY and Z_HUFFMAN_ONLY. Z_RLE is designed to be almost as fast as Z_HUFFMAN_ONLY, but give better compression for PNG image data. The strategy parameter only affects the compression ratio but not the correctness of the compressed output even if it is not set appropriately. Z_FIXED prevents the use of dynamic Huffman codes, allowing for a simpler decoder for special applications.

deflateInit2 returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_STREAM_ERROR if any parameter is invalid (such as an invalid method), or Z_VERSION_ERROR if the zlib library version (zlib_version) is incompatible with the version assumed by the caller (ZLIB_VERSION). msg is set to null if there is no error message. deflateInit2 does not perform any compression: this will be done by deflate().

*/

```
ZEXTERN int ZEXPORT deflateSetDictionary OF((z_streamp strm,
                                           const Bytef *dictionary,
                                           uInt dictLength));
```

/*

Initializes the compression dictionary from the given byte sequence without producing any compressed output. When using the zlib format, this

function must be called immediately after deflateInit, deflateInit2 or deflateReset, and before any call of deflate. When doing raw deflate, this function must be called either before any call of deflate, or immediately after the completion of a deflate block, i.e. after all input has been consumed and all output has been delivered when using any of the flush options Z_BLOCK, Z_PARTIAL_FLUSH, Z_SYNC_FLUSH, or Z_FULL_FLUSH. The compressor and decompressor must use exactly the same dictionary (see inflateSetDictionary).

The dictionary should consist of strings (byte sequences) that are likely to be encountered later in the data to be compressed, with the most commonly used strings preferably put towards the end of the dictionary. Using a dictionary is most useful when the data to be compressed is short and can be predicted with good accuracy; the data can then be compressed better than with the default empty dictionary.

Depending on the size of the compression data structures selected by deflateInit or deflateInit2, a part of the dictionary may in effect be discarded, for example if the dictionary is larger than the window size provided in deflateInit or deflateInit2. Thus the strings most likely to be useful should be put at the end of the dictionary, not at the front. In addition, the current implementation of deflate will use at most the window size minus 262 bytes of the provided dictionary.

Upon return of this function, strm->adler is set to the Adler32 value of the dictionary; the decompressor may later use this value to determine which dictionary has been used by the compressor. (The Adler32 value applies to the whole dictionary even if only a subset of the dictionary is actually used by the compressor.) If a raw deflate was requested, then the Adler32 value is not computed and strm->adler is not set.

deflateSetDictionary returns Z_OK if success, or Z_STREAM_ERROR if a parameter is invalid (e.g. dictionary being Z_NULL) or the stream state is inconsistent (for example if deflate has already been called for this stream or if not at a block boundary for raw deflate). deflateSetDictionary does not perform any compression: this will be done by deflate().

*/

```
ZEXTERN int ZEXPORT deflateCopy OF((z_streamp dest,
                                   z_streamp source));
```

/*

Sets the destination stream as a complete copy of the source stream.

This function can be useful when several compression strategies will be tried, for example when there are several ways of pre-processing the input data with a filter. The streams that will be discarded should then be freed by calling deflateEnd. Note that deflateCopy duplicates the internal compression state which can be quite large, so this strategy is slow and can consume lots of memory.

deflateCopy returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_STREAM_ERROR if the source stream state was inconsistent (such as zalloc being Z_NULL). msg is left unchanged in both source and destination.

*/

```
ZEXTERN int ZEXPORT deflateReset OF((z_streamp strm));
```

```
/*
```

This function is equivalent to deflateEnd followed by deflateInit, but does not free and reallocate all the internal compression state. The stream will keep the same compression level and any other attributes that may have been set by deflateInit2.

deflateReset returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent (such as zalloc or state being Z_NULL).

```
*/
```

```
ZEXTERN int ZEXPORT deflateParams OF((z_streamp strm,
                                     int level,
                                     int strategy));
```

```
/*
```

Dynamically update the compression level and compression strategy. The interpretation of level and strategy is as in deflateInit2. This can be used to switch between compression and straight copy of the input data, or to switch to a different kind of input data requiring a different strategy. If the compression level is changed, the input available so far is compressed with the old level (and may be flushed); the new level will take effect only at the next call of deflate().

Before the call of deflateParams, the stream state must be set as for a call of deflate(), since the currently available input may have to be compressed and flushed. In particular, strm->avail_out must be non-zero.

deflateParams returns Z_OK if success, Z_STREAM_ERROR if the source stream state was inconsistent or if a parameter was invalid, Z_BUF_ERROR if strm->avail_out was zero.

```
*/
```

```
ZEXTERN int ZEXPORT deflateTune OF((z_streamp strm,
                                   int good_length,
                                   int max_lazy,
                                   int nice_length,
                                   int max_chain));
```

```
/*
```

Fine tune deflate's internal compression parameters. This should only be used by someone who understands the algorithm used by zlib's deflate for searching for the best matching string, and even then only by the most fanatic optimizer trying to squeeze out the last compressed bit for their specific input data. Read the deflate.c source code for the meaning of the max_lazy, good_length, nice_length, and max_chain parameters.

deflateTune() can be called after deflateInit() or deflateInit2(), and returns Z_OK on success, or Z_STREAM_ERROR for an invalid deflate stream.

```
*/
```

```
ZEXTERN uLong ZEXPORT deflateBound OF((z_streamp strm,
                                       uLong sourceLen));
```

```
/*
```

deflateBound() returns an upper bound on the compressed size after

deflation of sourceLen bytes. It must be called after deflateInit() or deflateInit2(), and after deflateSetHeader(), if used. This would be used to allocate an output buffer for deflation in a single pass, and so would be called before deflate(). If that first deflate() call is provided the sourceLen input bytes, an output buffer allocated to the size returned by deflateBound(), and the flush value Z_FINISH, then deflate() is guaranteed to return Z_STREAM_END. Note that it is possible for the compressed size to be larger than the value returned by deflateBound() if flush options other than Z_FINISH or Z_NO_FLUSH are used.

*/

```
ZEXTERN int ZEXPORT deflatePending OF((z_streamp strm,
                                     unsigned *pending,
                                     int *bits));
```

/*

deflatePending() returns the number of bytes and bits of output that have been generated, but not yet provided in the available output. The bytes not provided would be due to the available output space having been consumed. The number of bits of output not provided are between 0 and 7, where they await more bits to join them in order to fill out a full byte. If pending or bits are Z_NULL, then those values are not set.

deflatePending returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent.

*/

```
ZEXTERN int ZEXPORT deflatePrime OF((z_streamp strm,
                                     int bits,
                                     int value));
```

/*

deflatePrime() inserts bits in the deflate output stream. The intent is that this function is used to start off the deflate output with the bits leftover from a previous deflate stream when appending to it. As such, this function can only be used for raw deflate, and must be used before the first deflate() call after a deflateInit2() or deflateReset(). bits must be less than or equal to 16, and that many of the least significant bits of value will be inserted in the output.

deflatePrime returns Z_OK if success, Z_BUF_ERROR if there was not enough room in the internal buffer to insert the bits, or Z_STREAM_ERROR if the source stream state was inconsistent.

*/

```
ZEXTERN int ZEXPORT deflateSetHeader OF((z_streamp strm,
                                         gz_headerp head));
```

/*

deflateSetHeader() provides gzip header information for when a gzip stream is requested by deflateInit2(). deflateSetHeader() may be called after deflateInit2() or deflateReset() and before the first call of deflate(). The text, time, os, extra field, name, and comment information in the provided gz_header structure are written to the gzip header (xflag is ignored -- the extra flags are set according to the compression level). The caller must assure that, if not Z_NULL, name and comment are terminated with a zero byte, and that if extra is not Z_NULL, that extra_len bytes are available there. If hcrc is true, a gzip header crc is included. Note that

the current versions of the command-line version of gzip (up through version 1.3.x) do not support header crc's, and will report that it is a "multi-part gzip file" and give up.

If deflateSetHeader is not used, the default gzip header has text false, the time set to zero, and os set to 255, with no extra, name, or comment fields. The gzip header is returned to the default state by deflateReset().

deflateSetHeader returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent.

*/

/*

```
ZEXTERN int ZEXPORT inflateInit2 OF((z_streamp strm,
                                     int windowBits));
```

This is another version of inflateInit with an extra parameter. The fields next_in, avail_in, zalloc, zfree and opaque must be initialized before by the caller.

The windowBits parameter is the base two logarithm of the maximum window size (the size of the history buffer). It should be in the range 8..15 for this version of the library. The default value is 15 if inflateInit is used instead. windowBits must be greater than or equal to the windowBits value provided to deflateInit2() while compressing, or it must be equal to 15 if deflateInit2() was not used. If a compressed stream with a larger window size is given as input, inflate() will return with the error code Z_DATA_ERROR instead of trying to allocate a larger window.

windowBits can also be zero to request that inflate use the window size in the zlib header of the compressed stream.

windowBits can also be -8..-15 for raw inflate. In this case, -windowBits determines the window size. inflate() will then process raw deflate data, not looking for a zlib or gzip header, not generating a check value, and not looking for any check values for comparison at the end of the stream. This is for use with other formats that use the deflate compressed data format such as zip. Those formats provide their own check values. If a custom format is developed using the raw deflate format for compressed data, it is recommended that a check value such as an Adler32 or a CRC32 be applied to the uncompressed data as is done in the zlib, gzip, and zip formats. For most applications, the zlib format should be used as is. Note that comments above on the use in deflateInit2() applies to the magnitude of windowBits.

windowBits can also be greater than 15 for optional gzip decoding. Add 32 to windowBits to enable zlib and gzip decoding with automatic header detection, or add 16 to decode only the gzip format (the zlib format will return a Z_DATA_ERROR). If a gzip stream is being decoded, strm->adler is a CRC32 instead of an Adler32.

inflateInit2 returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_VERSION_ERROR if the zlib library version is incompatible with the version assumed by the caller, or Z_STREAM_ERROR if the parameters are invalid, such as a null pointer to the structure. msg is set to null if

there is no error message. inflateInit2 does not perform any decompression apart from possibly reading the zlib header if present: actual decompression will be done by inflate(). (So next_in and avail_in may be modified, but next_out and avail_out are unused and unchanged.) The current implementation of inflateInit2() does not process any header information -- that is deferred until inflate() is called.

*/

```
ZEXTERN int ZEXPORT inflateSetDictionary OF((z_streamp strm,
                                             const Bytef *dictionary,
                                             uInt dictLength));
```

/*

Initializes the decompression dictionary from the given uncompressed byte sequence. This function must be called immediately after a call of inflate, if that call returned Z_NEED_DICT. The dictionary chosen by the compressor can be determined from the Adler32 value returned by that call of inflate. The compressor and decompressor must use exactly the same dictionary (see deflateSetDictionary). For raw inflate, this function can be called at any time to set the dictionary. If the provided dictionary is smaller than the window and there is already data in the window, then the provided dictionary will amend what's there. The application must insure that the dictionary that was used for compression is provided.

inflateSetDictionary returns Z_OK if success, Z_STREAM_ERROR if a parameter is invalid (e.g. dictionary being Z_NULL) or the stream state is inconsistent, Z_DATA_ERROR if the given dictionary doesn't match the expected one (incorrect Adler32 value). inflateSetDictionary does not perform any decompression: this will be done by subsequent calls of inflate().

*/

```
ZEXTERN int ZEXPORT inflateGetDictionary OF((z_streamp strm,
                                             Bytef *dictionary,
                                             uInt *dictLength));
```

/*

Returns the sliding dictionary being maintained by inflate. dictLength is set to the number of bytes in the dictionary, and that many bytes are copied to dictionary. dictionary must have enough space, where 32768 bytes is always enough. If inflateGetDictionary() is called with dictionary equal to Z_NULL, then only the dictionary length is returned, and nothing is copied. Similarly, if dictLength is Z_NULL, then it is not set.

inflateGetDictionary returns Z_OK on success, or Z_STREAM_ERROR if the stream state is inconsistent.

*/

```
ZEXTERN int ZEXPORT inflateSync OF((z_streamp strm));
```

/*

Skips invalid compressed data until a possible full flush point (see above for the description of deflate with Z_FULL_FLUSH) can be found, or until all available input is skipped. No output is provided.

inflateSync searches for a 00 00 FF FF pattern in the compressed data. All full flush points have this pattern, but not all occurrences of this pattern are full flush points.

inflateSync returns Z_OK if a possible full flush point has been found, Z_BUF_ERROR if no more input was provided, Z_DATA_ERROR if no flush point has been found, or Z_STREAM_ERROR if the stream structure was inconsistent. In the success case, the application may save the current value of total_in which indicates where valid compressed data was found. In the error case, the application may repeatedly call inflateSync, providing more input each time, until success or end of the input data.

*/

```
ZEXTERN int ZEXPORT inflateCopy OF((z_stream dest,
                                   z_stream source));
```

/*

Sets the destination stream as a complete copy of the source stream.

This function can be useful when randomly accessing a large stream. The first pass through the stream can periodically record the inflate state, allowing restarting inflate at those points when randomly accessing the stream.

inflateCopy returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_STREAM_ERROR if the source stream state was inconsistent (such as zalloc being Z_NULL). msg is left unchanged in both source and destination.

*/

```
ZEXTERN int ZEXPORT inflateReset OF((z_stream strm));
```

/*

This function is equivalent to inflateEnd followed by inflateInit, but does not free and reallocate all the internal decompression state. The stream will keep attributes that may have been set by inflateInit2.

inflateReset returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent (such as zalloc or state being Z_NULL).

*/

```
ZEXTERN int ZEXPORT inflateReset2 OF((z_stream strm,
                                     int windowBits));
```

/*

This function is the same as inflateReset, but it also permits changing the wrap and window size requests. The windowBits parameter is interpreted the same as it is for inflateInit2.

inflateReset2 returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent (such as zalloc or state being Z_NULL), or if the windowBits parameter is invalid.

*/

```
ZEXTERN int ZEXPORT inflatePrime OF((z_stream strm,
                                    int bits,
                                    int value));
```

/*

This function inserts bits in the inflate input stream. The intent is that this function is used to start inflating at a bit position in the

middle of a byte. The provided bits will be used before any bytes are used from next_in. This function should only be used with raw inflate, and should be used before the first inflate() call after inflateInit2() or inflateReset(). bits must be less than or equal to 16, and that many of the least significant bits of value will be inserted in the input.

If bits is negative, then the input stream bit buffer is emptied. Then inflatePrime() can be called again to put bits in the buffer. This is used to clear out bits leftover after feeding inflate a block description prior to feeding inflate codes.

inflatePrime returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent.

*/

ZEXTERN long ZEXPORT inflateMark OF((z_streamp strm));

/*

This function returns two values, one in the lower 16 bits of the return value, and the other in the remaining upper bits, obtained by shifting the return value down 16 bits. If the upper value is -1 and the lower value is zero, then inflate() is currently decoding information outside of a block. If the upper value is -1 and the lower value is non-zero, then inflate is in the middle of a stored block, with the lower value equaling the number of bytes from the input remaining to copy. If the upper value is not -1, then it is the number of bits back from the current bit position in the input of the code (literal or length/distance pair) currently being processed. In that case the lower value is the number of bytes already emitted for that code.

A code is being processed if inflate is waiting for more input to complete decoding of the code, or if it has completed decoding but is waiting for more output space to write the literal or match data.

inflateMark() is used to mark locations in the input data for random access, which may be at bit positions, and to note those cases where the output of a code may span boundaries of random access blocks. The current location in the input stream can be determined from avail_in and data_type as noted in the description for the Z_BLOCK flush parameter for inflate.

inflateMark returns the value noted above or -1 << 16 if the provided source stream state was inconsistent.

*/

ZEXTERN int ZEXPORT inflateGetHeader OF((z_streamp strm,
gz_headerp head));

/*

inflateGetHeader() requests that gzip header information be stored in the provided gz_header structure. inflateGetHeader() may be called after inflateInit2() or inflateReset(), and before the first call of inflate(). As inflate() processes the gzip stream, head->done is zero until the header is completed, at which time head->done is set to one. If a zlib stream is being decoded, then head->done is set to -1 to indicate that there will be no gzip header information forthcoming. Note that Z_BLOCK or Z_TREES can be used to force inflate() to return immediately after header processing is complete and before any actual data is decompressed.

The text, time, xflags, and os fields are filled in with the gzip header contents. hcrc is set to true if there is a header CRC. (The header CRC was valid if done is set to one.) If extra is not Z_NULL, then extra_max contains the maximum number of bytes to write to extra. Once done is true, extra_len contains the actual extra field length, and extra contains the extra field, or that field truncated if extra_max is less than extra_len. If name is not Z_NULL, then up to name_max characters are written there, terminated with a zero unless the length is greater than name_max. If comment is not Z_NULL, then up to comm_max characters are written there, terminated with a zero unless the length is greater than comm_max. When any of extra, name, or comment are not Z_NULL and the respective field is not present in the header, then that field is set to Z_NULL to signal its absence. This allows the use of deflateSetHeader() with the returned structure to duplicate the header. However if those fields are set to allocated memory, then the application will need to save those pointers elsewhere so that they can be eventually freed.

If inflateGetHeader is not used, then the header information is simply discarded. The header is always checked for validity, including the header CRC if present. inflateReset() will reset the process to discard the header information. The application would need to call inflateGetHeader() again to retrieve the header from the next gzip stream.

inflateGetHeader returns Z_OK if success, or Z_STREAM_ERROR if the source stream state was inconsistent.

*/

/*

```
ZEXTERN int ZEXPORT inflateBackInit OF((z_stream * strm, int windowBits,
                                         unsigned char FAR *window));
```

Initialize the internal stream state for decompression using inflateBack() calls. The fields zalloc, zfree and opaque in strm must be initialized before the call. If zalloc and zfree are Z_NULL, then the default library-derived memory allocation routines are used. windowBits is the base two logarithm of the window size, in the range 8..15. window is a caller supplied buffer of that size. Except for special applications where it is assured that deflate was used with small window sizes, windowBits must be 15 and a 32K byte window must be supplied to be able to decompress general deflate streams.

See inflateBack() for the usage of these routines.

inflateBackInit will return Z_OK on success, Z_STREAM_ERROR if any of the parameters are invalid, Z_MEM_ERROR if the internal state could not be allocated, or Z_VERSION_ERROR if the version of the library does not match the version of the header file.

*/

```
typedef unsigned (*in_func) OF((void FAR *,
                                z_const unsigned char FAR * FAR *));
typedef int (*out_func) OF((void FAR *, unsigned char FAR *, unsigned));
```

```
ZEXTERN int ZEXPORT inflateBack OF((z_streamp strm,
                                   in_func in, void FAR *in_desc,
                                   out_func out, void FAR *out_desc));
```

```
/*
```

inflateBack() does a raw inflate with a single call using a call-back interface for input and output. This is potentially more efficient than inflate() for file i/o applications, in that it avoids copying between the output and the sliding window by simply making the window itself the output buffer. inflate() can be faster on modern CPUs when used with large buffers. inflateBack() trusts the application to not change the output buffer passed by the output function, at least until inflateBack() returns.

inflateBackInit() must be called first to allocate the internal state and to initialize the state with the user-provided window buffer. inflateBack() may then be used multiple times to inflate a complete, raw deflate stream with each call. inflateBackEnd() is then called to free the allocated state.

A raw deflate stream is one with no zlib or gzip header or trailer. This routine would normally be used in a utility that reads zip or gzip files and writes out uncompressed files. The utility would decode the header and process the trailer on its own, hence this routine expects only the raw deflate stream to decompress. This is different from the normal behavior of inflate(), which expects either a zlib or gzip header and trailer around the deflate stream.

inflateBack() uses two subroutines supplied by the caller that are then called by inflateBack() for input and output. inflateBack() calls those routines until it reads a complete deflate stream and writes out all of the uncompressed data, or until it encounters an error. The function's parameters and return types are defined above in the in_func and out_func typedefs. inflateBack() will call in(in_desc, &buf) which should return the number of bytes of provided input, and a pointer to that input in buf. If there is no input available, in() must return zero--buf is ignored in that case--and inflateBack() will return a buffer error. inflateBack() will call out(out_desc, buf, len) to write the uncompressed data buf[0..len-1]. out() should return zero on success, or non-zero on failure. If out() returns non-zero, inflateBack() will return with an error. Neither in() nor out() are permitted to change the contents of the window provided to inflateBackInit(), which is also the buffer that out() uses to write from. The length written by out() will be at most the window size. Any non-zero amount of input may be provided by in().

For convenience, inflateBack() can be provided input on the first call by setting strm->next_in and strm->avail_in. If that input is exhausted, then in() will be called. Therefore strm->next_in must be initialized before calling inflateBack(). If strm->next_in is Z_NULL, then in() will be called immediately for input. If strm->next_in is not Z_NULL, then strm->avail_in must also be initialized, and then if strm->avail_in is not zero, input will initially be taken from strm->next_in[0 .. strm->avail_in - 1].

The in_desc and out_desc parameters of inflateBack() is passed as the first parameter of in() and out() respectively when they are called. These descriptors can be optionally used to pass any information that the caller-supplied in() and out() functions need to do their job.

On return, `inflateBack()` will set `strm->next_in` and `strm->avail_in` to pass back any unused input that was provided by the last `in()` call. The return values of `inflateBack()` can be `Z_STREAM_END` on success, `Z_BUF_ERROR` if `in()` or `out()` returned an error, `Z_DATA_ERROR` if there was a format error in the deflate stream (in which case `strm->msg` is set to indicate the nature of the error), or `Z_STREAM_ERROR` if the stream was not properly initialized. In the case of `Z_BUF_ERROR`, an input or output error can be distinguished using `strm->next_in` which will be `Z_NULL` only if `in()` returned an error. If `strm->next_in` is not `Z_NULL`, then the `Z_BUF_ERROR` was due to `out()` returning non-zero. (`in()` will always be called before `out()`, so `strm->next_in` is assured to be defined if `out()` returns non-zero.) Note that `inflateBack()` cannot return `Z_OK`.

*/

```
ZEXTERN int ZEXPORT inflateBackEnd OF((z_streamp strm));
```

/*

All memory allocated by `inflateBackInit()` is freed.

`inflateBackEnd()` returns `Z_OK` on success, or `Z_STREAM_ERROR` if the stream state was inconsistent.

*/

```
ZEXTERN uLong ZEXPORT zlibCompileFlags OF((void));
```

/* Return flags indicating compile-time options.

Type sizes, two bits each, 00 = 16 bits, 01 = 32, 10 = 64, 11 = other:

1.0: size of `uInt`

3.2: size of `uLong`

5.4: size of `voidpf` (pointer)

7.6: size of `z_off_t`

Compiler, assembler, and debug options:

8: `DEBUG`

9: `ASMV` or `ASMINF` -- use ASM code

10: `ZLIB_WINAPI` -- exported functions use the `WINAPI` calling convention

11: 0 (reserved)

One-time table building (smaller code, but not thread-safe if true):

12: `BUILDFIXED` -- build static block decoding tables when needed

13: `DYNAMIC_CRC_TABLE` -- build CRC calculation tables when needed

14,15: 0 (reserved)

Library content (indicates missing functionality):

16: `NO_GZCOMPRESS` -- `gz*` functions cannot compress (to avoid linking deflate code when not needed)

17: `NO_GZIP` -- deflate can't write gzip streams, and inflate can't detect and decode gzip streams (to avoid linking crc code)

18-19: 0 (reserved)

Operation variations (changes in library functionality):

20: `PKZIP_BUG_WORKAROUND` -- slightly more permissive inflate

21: `FASTEST` -- deflate algorithm with only one, lowest compression level

22,23: 0 (reserved)

The sprintf variant used by gzprintf (zero is best):

24: 0 = vs*, 1 = s* -- 1 means limited to 20 arguments after the format

25: 0 = *nprintf, 1 = *printf -- 1 means gzprintf() not secure!

26: 0 = returns value, 1 = void -- 1 means inferred string length returned

Remainder:

27-31: 0 (reserved)

*/

#ifndef Z_SOLO

/* utility functions */

/*

The following utility functions are implemented on top of the basic stream-oriented functions. To simplify the interface, some default options are assumed (compression level and memory usage, standard memory allocation functions). The source code of these utility functions can be modified if you need special options.

*/

ZEXTERN int ZEXPORT compress OF((Bytef *dest, uLongf *destLen,
const Bytef *source, uLong sourceLen));

/*

Compresses the source buffer into the destination buffer. sourceLen is the byte length of the source buffer. Upon entry, destLen is the total size of the destination buffer, which must be at least the value returned by compressBound(sourceLen). Upon exit, destLen is the actual size of the compressed buffer.

compress returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_BUF_ERROR if there was not enough room in the output buffer.

*/

ZEXTERN int ZEXPORT compress2 OF((Bytef *dest, uLongf *destLen,
const Bytef *source, uLong sourceLen,
int level));

/*

Compresses the source buffer into the destination buffer. The level parameter has the same meaning as in deflateInit. sourceLen is the byte length of the source buffer. Upon entry, destLen is the total size of the destination buffer, which must be at least the value returned by compressBound(sourceLen). Upon exit, destLen is the actual size of the compressed buffer.

compress2 returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_BUF_ERROR if there was not enough room in the output buffer, Z_STREAM_ERROR if the level parameter is invalid.

*/

ZEXTERN uLong ZEXPORT compressBound OF((uLong sourceLen));

/*

compressBound() returns an upper bound on the compressed size after

compress() or compress2() on sourceLen bytes. It would be used before a compress() or compress2() call to allocate the destination buffer.

```
*/
```

ZEXTERN int ZEXPORT uncompress OF((Bytef *dest, uLongf *destLen,
const Bytef *source, uLong sourceLen));

```
/*
```

Decompresses the source buffer into the destination buffer. sourceLen is the byte length of the source buffer. Upon entry, destLen is the total size of the destination buffer, which must be large enough to hold the entire uncompressed data. (The size of the uncompressed data must have been saved previously by the compressor and transmitted to the decompressor by some mechanism outside the scope of this compression library.) Upon exit, destLen is the actual size of the uncompressed buffer.

uncompress returns Z_OK if success, Z_MEM_ERROR if there was not enough memory, Z_BUF_ERROR if there was not enough room in the output buffer, or Z_DATA_ERROR if the input data was corrupted or incomplete. In the case where there is not enough room, uncompress() will fill the output buffer with the uncompressed data up to that point.

```
*/
```

/* gzip file access functions */

```
/*
```

This library supports reading and writing files in gzip (.gz) format with an interface similar to that of stdio, using the functions that start with "gz". The gzip format is different from the zlib format. gzip is a gzip wrapper, documented in RFC 1952, wrapped around a deflate stream.

```
*/
```

typedef struct gzFile_s *gzFile; /* semi-opaque gzip file descriptor */

```
/*
```

ZEXTERN gzFile ZEXPORT gzopen OF((const char *path, const char *mode));

Opens a gzip (.gz) file for reading or writing. The mode parameter is as in fopen ("rb" or "wb") but can also include a compression level ("wb9") or a strategy: 'f' for filtered data as in "wb6f", 'h' for Huffman-only compression as in "wb1h", 'R' for run-length encoding as in "wb1R", or 'F' for fixed code compression as in "wb9F". (See the description of deflateInit2 for more information about the strategy parameter.) 'T' will request transparent writing or appending with no compression and not using the gzip format.

"a" can be used instead of "w" to request that the gzip stream that will be written be appended to the file. "+" will result in an error, since reading and writing to the same gzip file is not supported. The addition of "x" when writing will create the file exclusively, which fails if the file already exists. On systems that support it, the addition of "e" when reading or writing will set the flag to close the file on an execve() call.

These functions, as well as gzip, will read and decode a sequence of gzip streams in a file. The append function of gzopen() can be used to create

such a file. (Also see `gzflush()` for another way to do this.) When appending, `gzopen` does not test whether the file begins with a gzip stream, nor does it look for the end of the gzip streams to begin appending. `gzopen` will simply append a gzip stream to the existing file.

`gzopen` can be used to read a file which is not in gzip format; in this case `gzread` will directly read from the file without decompression. When reading, this will be detected automatically by looking for the magic two-byte gzip header.

`gzopen` returns `NULL` if the file could not be opened, if there was insufficient memory to allocate the `gzFile` state, or if an invalid mode was specified (an 'r', 'w', or 'a' was not provided, or '+' was provided). `errno` can be checked to determine if the reason `gzopen` failed was that the file could not be opened.

*/

```
ZEXTERN gzFile ZEXPORT gzdopen OF((int fd, const char *mode));
```

/*

`gzdopen` associates a `gzFile` with the file descriptor `fd`. File descriptors are obtained from calls like `open`, `dup`, `creat`, `pipe` or `fileno` (if the file has been previously opened with `fopen`). The mode parameter is as in `gzopen`.

The next call of `gzclose` on the returned `gzFile` will also close the file descriptor `fd`, just like `fclose(fdopen(fd, mode))` closes the file descriptor `fd`. If you want to keep `fd` open, use `fd = dup(fd_keep); gz = gzdopen(fd, mode);`. The duplicated descriptor should be saved to avoid a leak, since `gzdopen` does not close `fd` if it fails. If you are using `fileno()` to get the file descriptor from a `FILE *`, then you will have to use `dup()` to avoid double-close()ing the file descriptor. Both `gzclose()` and `fclose()` will close the associated file descriptor, so they need to have different file descriptors.

`gzdopen` returns `NULL` if there was insufficient memory to allocate the `gzFile` state, if an invalid mode was specified (an 'r', 'w', or 'a' was not provided, or '+' was provided), or if `fd` is `-1`. The file descriptor is not used until the next `gz*` read, write, seek, or close operation, so `gzdopen` will not detect if `fd` is invalid (unless `fd` is `-1`).

*/

```
ZEXTERN int ZEXPORT gzbuffer OF((gzFile file, unsigned size));
```

/*

Set the internal buffer size used by this library's functions. The default buffer size is 8192 bytes. This function must be called after `gzopen()` or `gzdopen()`, and before any other calls that read or write the file. The buffer memory allocation is always deferred to the first read or write. Two buffers are allocated, either both of the specified size when writing, or one of the specified size and the other twice that size when reading. A larger buffer size of, for example, 64K or 128K bytes will noticeably increase the speed of decompression (reading).

The new buffer size also affects the maximum length for `gzprintf()`.

`gzbuffer()` returns 0 on success, or `-1` on failure, such as being called too late.

*/

ZEXTERN int ZEXPORT gzsetparams OF((gzFile file, int level, int strategy));

/*

Dynamically update the compression level or strategy. See the description of deflateInit2 for the meaning of these parameters.

gzsetparams returns Z_OK if success, or Z_STREAM_ERROR if the file was not opened for writing.

*/

ZEXTERN int ZEXPORT gzread OF((gzFile file, voidp buf, unsigned len));

/*

Reads the given number of uncompressed bytes from the compressed file. If the input file is not in gzip format, gzread copies the given number of bytes into the buffer directly from the file.

After reaching the end of a gzip stream in the input, gzread will continue to read, looking for another gzip stream. Any number of gzip streams may be concatenated in the input file, and will all be decompressed by gzread(). If something other than a gzip stream is encountered after a gzip stream, that remaining trailing garbage is ignored (and no error is returned).

gzread can be used to read a gzip file that is being concurrently written. Upon reaching the end of the input, gzread will return with the available data. If the error code returned by gzerror is Z_OK or Z_BUF_ERROR, then gzclearerr can be used to clear the end of file indicator in order to permit gzread to be tried again. Z_OK indicates that a gzip stream was completed on the last gzread. Z_BUF_ERROR indicates that the input file ended in the middle of a gzip stream. Note that gzread does not return -1 in the event of an incomplete gzip stream. This error is deferred until gzclose(), which will return Z_BUF_ERROR if the last gzread ended in the middle of a gzip stream. Alternatively, gzerror can be used before gzclose to detect this case.

gzread returns the number of uncompressed bytes actually read, less than len for end of file, or -1 for error.

*/

ZEXTERN int ZEXPORT gzwrite OF((gzFile file,
voidpc buf, unsigned len));

/*

Writes the given number of uncompressed bytes into the compressed file. gzwrite returns the number of uncompressed bytes written or 0 in case of error.

*/

ZEXTERN int ZEXPORTVA gzprintf Z_ARG((gzFile file, const char *format, ...));

/*

Converts, formats, and writes the arguments to the compressed file under control of the format string, as in fprintf. gzprintf returns the number of uncompressed bytes actually written, or 0 in case of error. The number of uncompressed bytes written is limited to 8191, or one less than the buffer size given to gzbuffer(). The caller should assure that this limit is not

exceeded. If it is exceeded, then `gzprintf()` will return an error (0) with nothing written. In this case, there may also be a buffer overflow with unpredictable consequences, which is possible only if zlib was compiled with the insecure functions `sprintf()` or `vsprintf()` because the secure `snprintf()` or `vsnprintf()` functions were not available. This can be determined using `zlibCompileFlags()`.

*/

ZEXTERN int ZEXPORT gzputs OF((gzFile file, const char *s));

/*

Writes the given null-terminated string to the compressed file, excluding the terminating null character.

gzputs returns the number of characters written, or -1 in case of error.

*/

ZEXTERN char * ZEXPORT gzgets OF((gzFile file, char *buf, int len));

/*

Reads bytes from the compressed file until `len-1` characters are read, or a newline character is read and transferred to `buf`, or an end-of-file condition is encountered. If any characters are read or if `len == 1`, the string is terminated with a null character. If no characters are read due to an end-of-file or `len < 1`, then the buffer is left untouched.

gzgets returns `buf` which is a null-terminated string, or it returns NULL for end-of-file or in case of error. If there was an error, the contents at `buf` are indeterminate.

*/

ZEXTERN int ZEXPORT gzputc OF((gzFile file, int c));

/*

Writes `c`, converted to an unsigned char, into the compressed file. `gzputc` returns the value that was written, or -1 in case of error.

*/

ZEXTERN int ZEXPORT gzgetc OF((gzFile file));

/*

Reads one byte from the compressed file. `gzgetc` returns this byte or -1 in case of end of file or error. This is implemented as a macro for speed. As such, it does not do all of the checking the other functions do. I.e. it does not check to see if `file` is NULL, nor whether the structure `file` points to has been clobbered or not.

*/

ZEXTERN int ZEXPORT gzungetc OF((int c, gzFile file));

/*

Push one character back onto the stream to be read as the first character on the next read. At least one character of push-back is allowed. `gzungetc()` returns the character pushed, or -1 on failure. `gzungetc()` will fail if `c` is -1, and may fail if a character has been pushed but not read yet. If `gzungetc` is used immediately after `gzopen` or `gzdopen`, at least the output buffer size of pushed characters is allowed. (See `gzbuffer` above.) The pushed character will be discarded if the stream is repositioned with `gzseek()` or `gzrewind()`.

*/


```
ZEXTERN int ZEXPORT gzflush OF((gzFile file, int flush));
```

```
/*
```

Flushes all pending output into the compressed file. The parameter flush is as in the deflate() function. The return value is the zlib error number (see function gzerror below). gzflush is only permitted when writing.

If the flush parameter is Z_FINISH, the remaining data is written and the gzip stream is completed in the output. If gzwrite() is called again, a new gzip stream will be started in the output. gzread() is able to read such concatenated gzip streams.

gzflush should be called only when strictly necessary because it will degrade compression if called too often.

```
*/
```

```
/*
```

```
ZEXTERN z_off_t ZEXPORT gzseek OF((gzFile file,
                                   z_off_t offset, int whence));
```

Sets the starting position for the next gzread or gzwrite on the given compressed file. The offset represents a number of bytes in the uncompressed data stream. The whence parameter is defined as in lseek(2); the value SEEK_END is not supported.

If the file is opened for reading, this function is emulated but can be extremely slow. If the file is opened for writing, only forward seeks are supported; gzseek then compresses a sequence of zeroes up to the new starting position.

gzseek returns the resulting offset location as measured in bytes from the beginning of the uncompressed stream, or -1 in case of error, in particular if the file is opened for writing and the new starting position would be before the current position.

```
*/
```

```
ZEXTERN int ZEXPORT gzrewind OF((gzFile file));
```

```
/*
```

Rewinds the given file. This function is supported only for reading.

gzrewind(file) is equivalent to (int)gzseek(file, 0L, SEEK_SET)

```
*/
```

```
/*
```

```
ZEXTERN z_off_t ZEXPORT gtell OF((gzFile file));
```

Returns the starting position for the next gzread or gzwrite on the given compressed file. This position represents a number of bytes in the uncompressed data stream, and is zero when starting, even if appending or reading a gzip stream from the middle of a file using gzdopen().

gtell(file) is equivalent to gzseek(file, 0L, SEEK_CUR)

```
*/
```

```
/*
ZEXTERN z_off_t ZEXPORT gzoffset OF((gzFile file));
```

Returns the current offset in the file being read or written. This offset includes the count of bytes that precede the gzip stream, for example when appending or when using `gzdopen()` for reading. When reading, the offset does not include as yet unused buffered input. This information can be used for a progress indicator. On error, `gzoffset()` returns -1.

```
*/
```

```
ZEXTERN int ZEXPORT gzeof OF((gzFile file));
```

```
/*
```

Returns true (1) if the end-of-file indicator has been set while reading, false (0) otherwise. Note that the end-of-file indicator is set only if the read tried to go past the end of the input, but came up short. Therefore, just like `feof()`, `gzeof()` may return false even if there is no more data to read, in the event that the last read request was for the exact number of bytes remaining in the input file. This will happen if the input file size is an exact multiple of the buffer size.

If `gzeof()` returns true, then the read functions will return no more data, unless the end-of-file indicator is reset by `gzcLEARerr()` and the input file has grown since the previous end of file was detected.

```
*/
```

```
ZEXTERN int ZEXPORT gzdirect OF((gzFile file));
```

```
/*
```

Returns true (1) if file is being copied directly while reading, or false (0) if file is a gzip stream being decompressed.

If the input file is empty, `gzdirect()` will return true, since the input does not contain a gzip stream.

If `gzdirect()` is used immediately after `gzopen()` or `gzdopen()` it will cause buffers to be allocated to allow reading the file to determine if it is a gzip file. Therefore if `gzbuffer()` is used, it should be called before `gzdirect()`.

When writing, `gzdirect()` returns true (1) if transparent writing was requested ("wT" for the `gzopen()` mode), or false (0) otherwise. (Note: `gzdirect()` is not needed when writing. Transparent writing must be explicitly requested, so the application already knows the answer. When linking statically, using `gzdirect()` will include all of the zlib code for gzip file reading and decompression, which may not be desired.)

```
*/
```

```
ZEXTERN int ZEXPORT gzclose OF((gzFile file));
```

```
/*
```

Flushes all pending output if necessary, closes the compressed file and deallocates the (de)compression state. Note that once file is closed, you cannot call `gzerror` with file, since its structures have been deallocated. `gzclose` must not be called more than once on the same file, just as `free` must not be called more than once on the same allocation.

`gzclose` will return `Z_STREAM_ERROR` if file is not valid, `Z_ERRNO` on a

```

file operation error, Z_MEM_ERROR if out of memory, Z_BUF_ERROR if the
last read ended in the middle of a gzip stream, or Z_OK on success.
*/

ZEXTERN int ZEXPORT gzclose_r OF((gzFile file));
ZEXTERN int ZEXPORT gzclose_w OF((gzFile file));
/*
    Same as gzclose(), but gzclose_r() is only for use when reading, and
    gzclose_w() is only for use when writing or appending. The advantage to
    using these instead of gzclose() is that they avoid linking in zlib
    compression or decompression code that is not used when only reading or only
    writing respectively. If gzclose() is used, then both compression and
    decompression code will be included the application when linking to a static
    zlib library.
*/

ZEXTERN const char * ZEXPORT gzerror OF((gzFile file, int *errnum));
/*
    Returns the error message for the last error which occurred on the given
    compressed file. errnum is set to zlib error number. If an error occurred
    in the file system and not in the compression library, errnum is set to
    Z_ERRNO and the application may consult errno to get the exact error code.

    The application must not modify the returned string. Future calls to
    this function may invalidate the previously returned string. If file is
    closed, then the string previously returned by gzerror will no longer be
    available.

    gzerror() should be used to distinguish errors from end-of-file for those
    functions above that do not distinguish those cases in their return values.
*/

ZEXTERN void ZEXPORT gzclearerr OF((gzFile file));
/*
    Clears the error and end-of-file flags for file. This is analogous to the
    clearerr() function in stdio. This is useful for continuing to read a gzip
    file that is being written concurrently.
*/

#endif /* !Z_SOLO */

        /* checksum functions */

/*
    These functions are not related to compression but are exported
    anyway because they might be useful in applications using the compression
    library.
*/

ZEXTERN uLong ZEXPORT Adler32 OF((uLong Adler, const Bytef *buf, uInt len));
/*
    Update a running Adler-32 checksum with the bytes buf[0..len-1] and
    return the updated checksum. If buf is Z_NULL, this function returns the
    required initial value for the checksum.

```

An Adler-32 checksum is almost as reliable as a CRC32 but can be computed much faster.

Usage example:

```
uLong Adler = Adler32(0L, Z_NULL, 0);

while (read_buffer(buffer, length) != EOF) {
    Adler = Adler32(Adler, buffer, length);
}
if (Adler != original_Adler) error();
*/

/*
ZEXTERN uLong ZEXPORT Adler32_combine OF((uLong Adler1, uLong Adler2,
                                          z_off_t len2));
```

Combine two Adler-32 checksums into one. For two sequences of bytes, seq1 and seq2 with lengths len1 and len2, Adler-32 checksums were calculated for each, Adler1 and Adler2. Adler32_combine() returns the Adler-32 checksum of seq1 and seq2 concatenated, requiring only Adler1, Adler2, and len2. Note that the z_off_t type (like off_t) is a signed integer. If len2 is negative, the result has no meaning or utility.

```
*/

ZEXTERN uLong ZEXPORT Crc32 OF((uLong Crc, const Bytef *buf, uInt len));
/*
```

Update a running CRC-32 with the bytes buf[0..len-1] and return the updated CRC-32. If buf is Z_NULL, this function returns the required initial value for the Crc. Pre- and post-conditioning (one's complement) is performed within this function so it shouldn't be done by the application.

Usage example:

```
uLong Crc = Crc32(0L, Z_NULL, 0);

while (read_buffer(buffer, length) != EOF) {
    Crc = Crc32(Crc, buffer, length);
}
if (Crc != original_Crc) error();
*/

/*
ZEXTERN uLong ZEXPORT Crc32_combine OF((uLong Crc1, uLong Crc2, z_off_t len2));
```

Combine two CRC-32 check values into one. For two sequences of bytes, seq1 and seq2 with lengths len1 and len2, CRC-32 check values were calculated for each, Crc1 and Crc2. Crc32_combine() returns the CRC-32 check value of seq1 and seq2 concatenated, requiring only Crc1, Crc2, and len2.

```
*/

/* various hacks, don't look :) */
```

```

/* deflateInit and inflateInit are macros to allow checking the zlib version
 * and the compiler's view of z_stream:
 */
ZEXTERN int ZEXPORT deflateInit_OF((z_stream *strm, int level,
                                   const char *version, int stream_size));
ZEXTERN int ZEXPORT inflateInit_OF((z_stream *strm,
                                   const char *version, int stream_size));
ZEXTERN int ZEXPORT deflateInit2_OF((z_stream *strm, int level, int method,
                                   int windowBits, int memLevel,
                                   int strategy, const char *version,
                                   int stream_size));
ZEXTERN int ZEXPORT inflateInit2_OF((z_stream *strm, int windowBits,
                                   const char *version, int stream_size));
ZEXTERN int ZEXPORT inflateBackInit_OF((z_stream *strm, int windowBits,
                                   unsigned char FAR *window,
                                   const char *version,
                                   int stream_size));

#define deflateInit(strm, level) \
    deflateInit_((strm), (level), ZLIB_VERSION, (int)sizeof(z_stream))
#define inflateInit(strm) \
    inflateInit_((strm), ZLIB_VERSION, (int)sizeof(z_stream))
#define deflateInit2(strm, level, method, windowBits, memLevel, strategy) \
    deflateInit2_((strm), (level), (method), (windowBits), (memLevel), \
                  (strategy), ZLIB_VERSION, (int)sizeof(z_stream))
#define inflateInit2(strm, windowBits) \
    inflateInit2_((strm), (windowBits), ZLIB_VERSION, \
                  (int)sizeof(z_stream))
#define inflateBackInit(strm, windowBits, window) \
    inflateBackInit_((strm), (windowBits), (window), \
                     ZLIB_VERSION, (int)sizeof(z_stream))

#ifndef Z_SOLO

/* gzgetc() macro and its supporting function and exposed data structure. Note
 * that the real internal state is much larger than the exposed structure.
 * This abbreviated structure exposes just enough for the gzgetc() macro. The
 * user should not mess with these exposed elements, since their names or
 * behavior could change in the future, perhaps even capriciously. They can
 * only be used by the gzgetc() macro. You have been warned.
 */
struct gzFile_s {
    unsigned have;
    unsigned char *next;
    z_off64_t pos;
};
ZEXTERN int ZEXPORT gzgetc_OF((gzFile file)); /* backward compatibility */
#ifdef Z_PREFIX_SET
# undef z_gzgetc
# define z_gzgetc(g) \
    ((g)->have ? ((g)->have--, (g)->pos++, *((g)->next)++) : gzgetc(g))
#else
# define gzgetc(g) \
    ((g)->have ? ((g)->have--, (g)->pos++, *((g)->next)++) : gzgetc(g))
#endif

```

```

#endif

/* provide 64-bit offset functions if _LARGEFILE64_SOURCE defined, and/or
 * change the regular functions to 64 bits if _FILE_OFFSET_BITS is 64 (if
 * both are true, the application gets the *64 functions, and the regular
 * functions are changed to 64 bits) -- in case these are set on systems
 * without large file support, _LFS64_LARGEFILE must also be true
 */
#ifndef Z_LARGE64
    ZEXTERN gzFile ZEXPORT gzopen64 OF((const char *, const char *));
    ZEXTERN z_off64_t ZEXPORT gzseek64 OF((gzFile, z_off64_t, int));
    ZEXTERN z_off64_t ZEXPORT gztell64 OF((gzFile));
    ZEXTERN z_off64_t ZEXPORT gzoffset64 OF((gzFile));
    ZEXTERN uLong ZEXPORT Adler32_combine64 OF((uLong, uLong, z_off64_t));
    ZEXTERN uLong ZEXPORT crc32_combine64 OF((uLong, uLong, z_off64_t));
#endif

#if !defined(ZLIB_INTERNAL) && defined(Z_WANT64)
# ifdef Z_PREFIX_SET
#   define z_gzopen z_gzopen64
#   define z_gzseek z_gzseek64
#   define z_gztell z_gztell64
#   define z_gzoffset z_gzoffset64
#   define z_adler32_combine z_adler32_combine64
#   define z_crc32_combine z_crc32_combine64
# else
#   define gzopen gzopen64
#   define gzseek gzseek64
#   define gztell gztell64
#   define gzoffset gzoffset64
#   define adler32_combine adler32_combine64
#   define crc32_combine crc32_combine64
# endif
# ifndef Z_LARGE64
    ZEXTERN gzFile ZEXPORT gzopen64 OF((const char *, const char *));
    ZEXTERN z_off_t ZEXPORT gzseek64 OF((gzFile, z_off_t, int));
    ZEXTERN z_off_t ZEXPORT gztell64 OF((gzFile));
    ZEXTERN z_off_t ZEXPORT gzoffset64 OF((gzFile));
    ZEXTERN uLong ZEXPORT Adler32_combine64 OF((uLong, uLong, z_off_t));
    ZEXTERN uLong ZEXPORT crc32_combine64 OF((uLong, uLong, z_off_t));
# endif
# else
    ZEXTERN gzFile ZEXPORT gzopen OF((const char *, const char *));
    ZEXTERN z_off_t ZEXPORT gzseek OF((gzFile, z_off_t, int));
    ZEXTERN z_off_t ZEXPORT gztell OF((gzFile));
    ZEXTERN z_off_t ZEXPORT gzoffset OF((gzFile));
    ZEXTERN uLong ZEXPORT Adler32_combine OF((uLong, uLong, z_off_t));
    ZEXTERN uLong ZEXPORT crc32_combine OF((uLong, uLong, z_off_t));
# endif

#else /* Z_SOLO */

    ZEXTERN uLong ZEXPORT Adler32_combine OF((uLong, uLong, z_off_t));
    ZEXTERN uLong ZEXPORT crc32_combine OF((uLong, uLong, z_off_t));

```

```
#endif /* !Z_SOLO */

/* hack for buggy compilers */
#if !defined(ZUTIL_H) && !defined(NO_DUMMY_DECL)
    struct internal_state {int dummy;};
#endif

/* undocumented functions */
ZEXTERN const char * ZEXPORT zError      OF((int));
ZEXTERN int          ZEXPORT inflateSyncPoint OF((z_streamp));
ZEXTERN const z_crc_t FAR * ZEXPORT get_crc_table  OF((void));
ZEXTERN int          ZEXPORT inflateUndermine OF((z_streamp, int));
ZEXTERN int          ZEXPORT inflateResetKeep OF((z_streamp));
ZEXTERN int          ZEXPORT deflateResetKeep OF((z_streamp));
#if defined(_WIN32) && !defined(Z_SOLO)
ZEXTERN gzFile       ZEXPORT gzopen_w OF((const wchar_t *path,
                                         const char *mode));
#endif
#if defined(STDC) || defined(Z_HAVE_STDARG_H)
# ifndef Z_SOLO
ZEXTERN int          ZEXPORTVA gzvprintf Z_ARG((gzFile file,
                                                const char *format,
                                                va_list va));
# endif
#endif
#endif

#ifdef __cplusplus
}
#endif

#endif /* ZLIB_H */
```

1.154 zram-config

1.154.1 Available under license

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL

assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the

source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- * a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- * b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- * c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the

work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

* d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

* a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

* b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

* c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

* d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

* e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the

modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- * a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- * b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- * c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- * d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- * e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- * f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the

right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from

time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>  
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.