GNU `inetutils`

GNU networking utilities
for version 1.9.3, 31 March 2015

Alain Magloire et al.

This manual documents version 1.9.3 of the GNU networking utilities.

Short Contents

Table of Contents

1 Introduction

The GNU Network Utilities is a distribution of common networking utilities and servers, including for example ping, traceroute and ftp.

This manual is a work in progress: many sections make no attempt to explain basic concepts in a way suitable for novices. Thus, if you are interested, please get involved in improving this manual. The entire GNU community will benefit.

Please report bugs to `bug-inetutils@gnu.org`. Remember to include the version number, machine architecture, input files, and any other information needed to reproduce the bug: your input, what you expected, what you got, and why it is wrong. Diffs are welcome, but please include a description of the problem as well, since this is sometimes difficult to infer.

The individual utilities were originally derived from the 4.4BSDLite2 distribution, although some of them have more or less been rewritten. What you are reading now is the authoritative and complete documentation for these utilities; the man pages are now automatically generated.

Many features were integrated from NetBSD, OpenBSD, FreeBSD and GNU/Linux, the merges were done by a group of dedicated hackers (in no particular order): Jeff Bailey, Marcus Brinkmann, Michael Vogt, Bernhard Rosenkraenzer, Kaveh R. Ghazi, NIIBE Yutaka, Nathan Neulinger, Jeff Smith, Dan Stromberg, David O'Shea, Frederic Goudal, Gerald Combs, Joachim Gabler, Marco D'Itri, Sergey Poznyakoff, and many more.

2 Common options

Certain options are available in all these programs. Rather than writing identical descriptions for each of the programs, they are described here. (In fact, every GNU program accepts, or should accept, these options.)

Many of these programs take arbitrary strings as arguments. In those cases, `--help` and `--version` are taken as these options only if there is one and exactly one command line argument.

`--help` Print a usage message, listing all available options, then exit successfully.

`--usage` Print a condensed usage message, displaying all available options formatted like a command line call, then exit successfully.

`--version`
Print the version number, then exit successfully.

`--` Delimit the option list. Later arguments, if any, are treated as operands even if they begin with ‘`-`’.

2.1 Exit status

Nearly every command invocation yields an integral *exit status* that can be used to change how other commands work. For the vast majority of commands, an exit status of zero indicates success. Failure is indicated by a nonzero value — typically ‘`1`’, though it may differ on unusual platforms, as POSIX requires only that it be nonzero.

3 dnsdomainname: Show DNS domain name

`dnsdomainname` is a program to show the domain part of the system's fully qualified domain name. For example, if the FQDN of the system is `name.example.org` the command will show `example.org`. The output is not necessarily related to the NIS/YP domain name.

The tool uses gethostname to get the host name of the system and then getaddrinfo to resolve it into a canonical name. The domain part of the canonical name is shown, i.e., the part after the first period (`.`) of the official name.

Synopsis:

```
dnsdomainname [option...]
```

There is no command specific option.

4 hostname: Show or set system host name.

`hostname` is a program to show or to set the name of a host system.

Synopsis:

```
hostname [option...]
hostname name
```

where *name* is the name to be used by the running host.

4.1 Command line options

`-a`
`--aliases`
Get alias names.

`-d`
`--domain` Get DNS domain name.

`-f`
`--fqdn`
`--long` Get DNS host name or Fully Qualified Domain Name.

`-F` *`file`*
`--file=`*`file`*
Set host name or NIS domain name from FILE.

`-i`
`--ip-addresses`
Get addresses for the host name.

`-s`
`--short` Get short host name.

`-y`
`--yp`
`--nis` Get NIS/YP domain name.

5 `logger`: Send messages to system log

`logger` is a program to send entries to system log. It provides a shell command interface similar to the system log module. For background information, see Section "Syslog" in *The GNU C Library Reference Manual*.

Synopsis:

```
logger [option...] [message]
```

5.1 Command line options

`-4`
`--ipv4`
Use IPv4 as transport when logging to a host. The default behaviour is to use whatever IP version that matches the host.

`-6`
`--ipv6`
Use IPv6 as transport when logging to a host. The option is present also on systems without support for IPv6, but will then issue a warning and then fall back to IPv4 when delivering the message.

Both options are most influencial when the target host is named using a symbolic name, but numerical addresses for host or source must also match if either of `--ipv4` or `--ipv6` is stated.

`-f` *`file`*
`--file=`*`file`*
Log the content of the specified file. If *file* is '`-`' then standard input is assumed.

`-h` *`host`*
`--host=`*`host`*
Send messages to the given host or socket. The *host* argument can be either a local UNIX socket name (containing a slash '`/`'), or be of the form

```
host[:port]
```

where *host* is the remote host name or IP address, and the optional *port* is a decimal port number or symbolic service name from `/etc/services`. If *port* is not specified, the port number corresponding to the '`syslog`' service is used. If a numerical IPv6 address is given without a port specification, then the address must be enclosed within brackets (like [::1]).

`-i[`*`pid`*`]`
`--id=[`*`pid`*`]`
Add process ID to each message. If *pid* is not supplied, use the process ID of the logger process with each line. Notice, that *pid* is an optional argument. When supplied to the `-i` option, it must follow the '`i`' letter immediately, without any separating whitespace. When supplied to the `--id` form, it must be separated from it by exactly one equals sign.

`-p` *`priority`*
`--priority=`*`priority`*
Enter the message with the specified priority. The priority may be specified numerically or as a '`facility.level`' pair. For example, `-p local3.info` logs

the message at the informational level in the '`local3`' facility. The default is '`user.notice`'.

The actual list of supported facilities and levels is system specific.

`-s`
`--stderr` Log the message to standard error, as well as to the system log.

`-S` *`addr`*
`--source=`*`addr`*

Supply the source IP address for INET connections. This option is useful in conjunction with `--host` (see above). The kind of address specified here (IPv4 or IPv6) will propagate to influence the resolution of the host address, if it is a symbolic name.

`-t` *`tag`*
`--tag=`*`tag`*

Mark every line in the log with the specified tag.

`-u` *`socket`*
`--unix=`*`socket`*

Send messages to the given local UNIX socket. The *socket* argument can be either an absolute path (starting with a slash '`/`'), or a relative path understood relative to the current working directory.

The options are followed by the message which should be written to the log. If not specified, and the `-f` flag is not provided, standard input is logged.

5.2 Examples

The following examples illustrate the usage of the `logger` command:

1. Log the message '`System rebooted`' to the local syslog. Use default facility and priority:

```
logger System rebooted
```

2. Run command and send its error output to the channel '`local0.err`'. Mark each message with tag '`cmd`':

```
command 2>&1 | logger -p local0.err -t cmd
```

3. Log each line from file `warnings` to channel '`daemon.warn`' on host '`logger.runasimi.org`', using the source IP '`10.10.10.1`':

```
logger -p daemon.warn -h logger.runasimi.org -S 10.10.10.1  \
       --file warnings
```

6 ping: Packets to network hosts

ping uses ICMP datagrams to provoke a response from the chosen destination host, mainly intending to probe whether it is alive.

The used datagram, of type ECHO_REQUEST, contains some header information and some additional payload, usually a time stamp. By a suitable choice of payload, different host or router properties are detectable, as the emitted datagram travels to its destination.

Synopsis:

```
ping [option...] host
```

Sending echo requests is the standard use of ping, but by far not the only use case.

6.1 Command line options

Selection of packet type is handled by these first options:

--address
: Send ICMP_ADDRESS packets, thus requesting the address netmask in use by the targetted host.

--echo
: Send ICMP_ECHO requests. This is the default action.

--mask
: Identical to --address.

--timestamp
: Send ICMP_TIMESTAMP packets, thereby requesting a timed response from the targetted host.

In successful cases three time values are returned. All are expected to state the number of milliseconds since midnight UTC. The first of these, 'icmp_otime', contains the original time of sending the request. Then comes 'icmp_rtime', the time of reception by the target, and finally, 'icmp_ttime', the time of transmitting an answer back to the originator.

-t *type*
--type=*type*
: Send *type* packets. Accepted values are 'address', 'echo', 'mask', and 'timestamp'.

The following options are available for all packet types:

-c *n*
--count=*n*
: Stop after sending and receiving answers to a total of *n* packets.

-d
--debug
: Set the SO_DEBUG option on the socket being used.

-i *n*
--interval=*n*
: Wait *n* seconds until sending next packet. The default is to wait for one second between packets. This option is incompatible with the option -f.

`-n`
`--numeric`

Numeric output only. No attempt will be made to resolve symbolic names for host addresses.

`-r`
`--ignore-routing`

Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly attached network, an error is returned. This option can be used to ping a local host through an interface that has no route through it (e.g., after the interface was dropped by `routed`).

`-T` *num*
`--tos=`*num*

Set type-of-service, TOS field, to *num* on transmitted packets.

`--ttl=`*n*

Set the specified number *n* as value of time-to-live when transmitting packets. Acceptable values are 1 to 255, inclusive.

`-v`
`--verbose`

Produce more verbose output, giving more statistics.

`-w` *n*
`--timeout=`*n*

Stop after *n* seconds.

`-W` *n*
`--linger=`*n*

Maximum number of seconds *n* to wait for a response.

Finally, these last options are relevant only for sending echo requests, allowing many variations in order to detect various peculiarities of the targeted host, or the intermediary routers for that matter.

`-f`
`--flood`

Flood ping. Outputs packets as fast as they come back or one hundred times per second, whichever is more. For every ECHO_REQUEST packet sent, a period '`.`' is printed, while for every ECHO_REPLY received in reply, a backspace is printed. This provides a rapid display of how many packets are being dropped. Only the super-user may use this option. This can be very hard on a network and should be used with caution.

`--ip-timestamp=`*flag*

Include IP option Timestamp in transmitted packets. The value *flag* is either '`tsonly`', which only records up to nine time stamps, or '`tsaddr`', which records IP addresses as well as time stamps, but for at most four hosts.

`-l` *n*
`--preload=`*n*

If *n* is specified, ping sends that many packets as fast as possible before falling into its normal mode of operation.

`-p` *pat*
`--pattern=`*pat*
: You may specify up to 16 pad bytes to fill out the packet you send. This is useful for diagnosing data-dependent problems in a network. For example, `-p ff` will cause the sent packet to be filled with all ones.

`-q`
`--quiet`
: Do not print timing for each transmitted packet.

`-R`
`--route`
: Record route. Includes the `RECORD_ROUTE` field in the ECHO_REQUEST packet and displays the route buffer on returned packets. Note that the IP header is only large enough for nine such routes. Many hosts ignore or discard this option.

`-s` *n*
`--size=`*n*
: Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes, taking the 8 bytes of ICMP header data into account.

6.2 Using ping for network fault isolation

When using `ping` for fault isolation, it should first be run on the local host, to verify that the local network interface is up and running. Then, hosts and gateways further and further away should be pinged. Round-trip times and packet loss statistics are computed. If duplicate packets are received, they are not included in the packet loss calculation, although the round trip time of these packets is used in calculating the minimum/average/maximum round-trip time numbers. When the specified number of packets have been sent (and received) or if the program is terminated with a '`SIGINT`', a brief summary is displayed.

This program is intended for use in network testing, measurement and management. Because of the load it can impose on the network, it is unwise to use ping during normal operations or from automated scripts.

6.3 Duplicate and damaged packets

Ping will report duplicate and damaged packets. Duplicate packets should never occur, and seem to be caused by inappropriate link-level retransmissions. Duplicates may occur in many situations and are rarely (if ever) a good sign, although the presence of low levels of duplicates may not always be cause for alarm.

Damaged packets are obviously serious cause for alarm and often indicate broken hardware somewhere in the ping packet's path (in the network or in the hosts).

6.4 Trying different data patterns

The (inter)network layer should never treat packets differently depending on the data contained in the data portion. Unfortunately, data-dependent problems have been known to sneak into networks and remain undetected for long periods of time. In many cases the particular pattern that will have problems is something that doesn't have sufficient "transitions", such as all ones or all zeros, or a pattern right at the edge, such as almost all zeros. It isn't necessarily enough to specify a data pattern of all zeros (for example) on the command line because the pattern that is of interest is at the data link level, and the relationship between what you type and what the controllers transmit can be complicated.

This means that if you have a data-dependent problem you will probably have to do a lot of testing to find it. If you are lucky, you may manage to find a file that either can't be sent across your network or that takes much longer to transfer than other similar length files. You can then examine this file for repeated patterns that you can test using the `-p` option of ping.

6.5 TTL details

The TTL field, *Time To Live*, of an IP packet represents the maximum number of IP routers that the packet can go through before being discarded. In current practice you can expect each router on the Internet to decrement the TTL field by exactly one.

The TCP/IP specification states that the TTL field of a new TCP packet should be set to 60, but many systems use smaller values (4.3BSD used 30 and 4.2BSD used 15).

The maximum possible value of this field is 255, and most UNIX systems set the TTL field of ICMP (type `ECHO_REQUEST`) packets to 255. This is why you will find you can ping some hosts, but not reach them with `telnet` or `ftp`.

During normal operation, `ping` prints the TTL value for every packet it receives. When a remote system receives an ICMP packet, it can do one of three things to the TTL field in its response packet:

- Not to change it. This is what Berkeley UNIX systems did before the 4.3BSD-Tahoe release. In this case the TTL value in the received packet will be 255 minus the number of routers in the round-trip path.
- Set it to 255. This is what current Berkeley UNIX systems do. In this case the TTL value in the received packet will be 255 minus the number of routers in the path from the remote system to the pinging host.
- Set it to some other value. Some machines use the same value for ICMP packets that they use for TCP packets, for example either 30 or 60. Others may use completely arbitrary values.

6.6 Further observations

Many hosts and gateways ignore the `RECORD_ROUTE` field, since the maximum IP header length is far to small to hold all the routes. There is not much that can be done about this.

Flood pinging is not recommended in general, and flood pinging the broadcast address should only be done under very controlled conditions.

Some BSD variants offer a kernel setting to inhibit all replies to ICMP_MASKREQ packets, but in general, Unices are designed either to answer the request with a valid netmask, or to drop the request, causing `ping` to wait for a timeout condition.

7 `ping6`: Packets to IPv6 network hosts

`ping6` uses ICMPv6 datagrams to get a response from the chosen destination host. The most common use is to probe whether the remote system is responsive. Observe that this program only uses IPv6 datagrams.

Each datagram, of type `ECHO_REQUEST`, carries some header information and some additional payload, usually a time stamp. Making a suitable choice of payload, it is possible to probe different host or router properties on the way as the emitted datagram travels to its destination.

Synopsis:

```
ping6 [option...] host
```

Sending simple, timed echo requests is the standard use of `ping6`, but is by far not the only use case.

This command is a close parallel to `ping`, except that it handles IPv6 and is thus not able to handle peculiarities of IPv4.

7.1 Command line options

`-c` *n*
`--count=`*n*
: Stop after sending and receiving answers to a total of *n* packets.

`-d`
`--debug`
: Set the SO_DEBUG option on the socket being used.

`-f`
`--flood`
: Flood ping. Outputs packets as fast as they come back, or one hundred times per second, whichever is more. For every ECHO_REQUEST packet sent, a period '`.`' is printed, while for every ECHO_REPLY received in reply, a backspace is printed.

This provides a rapid display of how many packets are being dropped. Only the super-user may use this option. This mode can be very hard on a network. It should be used with caution!

`--hoplimit=`*n*
: Limit maximal distance to *n*. Acceptable values are 1 to 255, inclusive.

`-i` *n*
`--interval=`*n*
: Wait *n* seconds until sending next packet. The default is to wait for one second between packets. This option is incompatible with the option `-f`.

`-l` *n*
`--preload=`*n*
: Sends *n* packets as fast as possible before falling back to the normal mode of operation.

`-n`
`--numeric`
: Numeric output only. No attempt will be made to resolve symbolic names for host addresses.

`-p` *`pattern`*
`--pattern=`*`pattern`*
: Up to 16 hexadecimal pad bytes are given as *pattern*. These are use for filling out the packets you send. This option is useful for diagnosing data-dependent problems within a network. As an example, `-p ff` will cause the sent packets to have payloads with every bit set to one.

`-q`
`--quiet` Do not print timing result of each transmitted packet.

`-r`
`--ignore-routing`
: Bypass the normal routing tables and send directly to a host on an attached network. If the host is not on a directly attached network, an error is returned. This option can be used to ping a local host through an interface, for which there is no assigned route, such as when the interface was dropped by `routed`.

`-s` *`n`*
`--size=`*`n`* Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes, taking the 8 bytes of ICMP header data into account.

`-T` *`num`*
`--tos=`*`num`*
: Set the traffic class to *num* on transmitted packets.

`--ttl=`*`n`* Synonym for `--hoplimit`.

`-v`
`--verbose`
: Produce more verbose output, giving more statistics.

`-w` *`n`*
`--timeout=`*`n`*
: Stop after *n* seconds.

The documentation of `ping` provides several pieces of information, and discussions, relevant to the use of `ping6`. Keep in mind, though, that the differing address family causes some discrepancy. See Chapter 6 [ping invocation], page 7.

8 traceroute: Trace the route to a host

`traceroute` prints a trace of the route IP packets are travelling to a remote host.

Synopsis:

```
traceroute [option...] host
```

8.1 Command line options

`-f num`
`--first-hop=num`
Set the initial hop distance to *num*, instead of the default 1. This immediately allows probing packets to sense routing properties closer to the target host, skipping routers close to the local host. Quicker analysis of problems known to lie at some routing distance is the outcome.

`-g gates`
`--gateways=gates`
Set intermediary hosts used in loose source routing. The argument *gates* is a list of gateways, using spaces, commata, or semicola as separators. These hosts must be traversed in the given order before the intended host receives any datagram. At most eight host names or addresses may be specified. Multiple uses of `-g` produce a concatenated list.

`-I`
`--icmp` Use ICMP ECHO datagrams for probing the remote host.

`-m num`
`--max-hop=num`
Set the maximum time-to-live allowed for probing. In other words, stop probing when the hop distance is in excess of *num*. The default limit is 64.

`-M method`
`--type=method`
Use *method* as carrier packets for traceroute operations. Supported choices are '`icmp`' and '`udp`', where '`udp`' is the default type.

`-p port`
`--port=port`
Set destination port of target to *port*. The default value is 33434.

`-q num`
`--tries=num`
Send a total of *num* probe packets per hop, defaulting to 3.

`--resolve-hostnames`
Attempt to resolve all addresses as hostnames.

`-t num`
`--tos=num`
Set type-of-service, TOS field, to *num* on transmitted packets.

`-w` *num*
`--wait=`*num*

Set timeout in seconds, within which a returning response packet is accepted as such. Default waiting time is three seconds.

8.2 Diagnostic tokens

During execution, `traceroute` sends three datagrams for each value for the TTL field, printing a diagnostic line of output for these. The TTL field is then steadily increased until the intended host responds, or some intermediary gateway returns a datagram to the effect that the target cannot be reached due to one reason or another.

Each line of output displays a sequence number, followed by diagnostic annotation. Any responding host has its address printed without repetition, together with a measured timing. In case there is no response within a time period of three seconds, an asterisque '`*`' is printed.

When an intermediate router responds with an exceptional state, the time elapsed since emitting the original datagram is printed, followed by an additional short hand hint of the reason:

'`!F`'	Fragmentation needed by gateway.
'`!H`'	Host not reachable from gateway.
'`!N`'	Network not reachable from gateway.
'`!P`'	Protocol not usable at host, or within network.
'`!S`'	Source routing failed at gateway.
'`!T`'	Host or network not reachable for stated type of service, TOS.
'`!U`'	Isolated host, not reachable.
'`!X`'	Forbidden by remote administration.

9 whois: User interface to WHOIS data bases.

The functionality of a world wide Internet is dependent on stored node information of different kinds. Registrars keep much relevant material in WHOIS data bases. This utility `whois` is able to query those sources for general and for particular properties of most domains.

For many domains there are names of suitable data base servers hard coded into `whois`, ready to query for domain relevant information. Since servers' names do change from time to time, this utility might occasionally need some guidance using a suitable command line option.

Synopsis:

```
whois [OPTION...] OBJECT...
```

9.1 Command line options

`-a` Search all data bases.

`-F` Fast and raw output. Implies `-r`.

`-g` *source*`:`*first-last*
Find updates for an object from provider *source*, starting from the version with serial key *first*, and ending with serial key *last*.

`-h` *host*
`--server=`*host*
Connect to server *host*.

`-H` Hide legal disclaimers.

`-i` *attr*`[,`*attr2*`...]`
Do an inverse lookup for specified attributes. Use a comma separated list for multiple attributes.

`-l` One level less specific lookup. Applies to RPSL only.

`-L` Find all less specific matches.

`-m` Find more specific matches, one level deeper.

`-M` Find all more specific matches.

`-p` *port* Connect to server port *port*.

`-q {version|sources}`
Query specified server info. Applies to RPSL only.

`-r` Turn off recursive lookups.

`-R` Force output to show local copy of the domain object, even if it contains a referral.

`-s` *source*`[,`*source2*`...]`
Search the data base at *source*. A comma separated list queries multiple providers.

`-S` Tell server to refrain from syntactic sugar.

`-t` *`type`* Request a template for objects of type *type*. Use the value '`all`' for a list of possible types.

`-T` *`type`*`[,`*`type2`*`...]`
Search only for objects of type *type*. A comma separated list allows for multiple types.

`-V`
`--verbose`
Verbosely explain all actions taken.

`-x` Search only for exact matches. Applicable only to RPSL.

9.2 Environment variables

`whois` holds an internal list of information servers and their assigned data bases. Queries are examined against this list to select the most plausible server, but the hint can always be overruled on the command line by use of the option `-h`. If neither of these have a say, then the default server to ask is '`whois.internic.net`', but this name is in turn overruled by a server name in the environment variable `WHOIS_SERVER`.

`LANG` When the server '`whois.nic.ad.jp`' is queried, and only then, any non-Japanese locale in `LANG` will ask the server to reply with English text, not Japanese.

`WHOIS_HIDE`
When set, the effect on `whois` is as if the option `-H` had been given.

`WHOIS_SERVER`
Data base server to query when internal hinting is inconclusive. When unset, '`whois.internic.net`' is used as default server.

10 ftp: FTP client

ftp is the user interface to FTP, the File Transfer Protocol. The program allows a user to transfer files to and from a remote network site.

Synopsis:

```
ftp [option...] [host [port]]
pftp [option...] [host [port]]
```

The alternate name pftp is starting in passive mode, but is otherwise identical to ftp.

The client host with which ftp is to communicate may be specified on the command line. If this is done, ftp will immediately attempt to establish a connection to the FTP server running on that host. Otherwise, the program will start a command interpreter and will await further instructions from the user. Commands can either be entered interactively, or piped as a batched job read from standard input. ftp is able to distinguish between these two modes of operation.

10.1 Command line options

Many command line options have counterparts among the commands handled by the internal interpreter.

-4
--ipv4 Initially set addressing to IPv4 only.

-6
--ipv6 Initially set addressing to IPv6 only.

-A
--active Enable active mode transfer. Default mode for ftp.

-d
--debug Enable debugging output and possibly also socket debugging.

-e
--no-edit
Disables the editing of commands. This is default setting for batch mode, without a TTY.

-g
--no-glob
Disables file name globbing.

-i
--no-prompt
Turns off interactive prompting during multiple file transfers.

-N *netrc*
--netrc=*netrc*
Set a preferred location of the .netrc file, thus overriding any environment setting in NETRC, as well as the default location $HOME/.netrc, see Section 10.7 [The .netrc file], page 27.

`-n`
`--no-login`
Restrains `ftp` from attempting *auto-login* upon initial connection. If auto-login is enabled, `ftp` will check the `.netrc` (see Section 10.7 [The .netrc file], page 27) file in the user's home directory for an entry describing an account on the remote machine. If no entry exists, `ftp` will prompt for the remote machine login name (default is the user identity on the local machine), and, if necessary, prompt for a password and an account with which to login.

`-p`
`--passive`
Enable passive mode transfer. Default mode when invoked as `pftp`.

`--prompt[=`*prompt*`]`
Print a command-line prompt, even if not on a tty. If *prompt* is supplied, its value is used instead of the default '`ftp> `'. Notice, that the argument is optional.

`-t`
`--trace` Enable packet tracing (not implemented).

`-v`
`--verbose`
Start in verbose mode, printing informational messages. This is default for interactive mode.

10.2 Commands interpreted by ftp

When `ftp` is awaiting commands from the user, a prompt is displayed. The default string is '`ftp>`', but it can been changed with a command line option, perhaps to enhance uniqueness while recording a session.

Be aware that correct execution of many commands depends upon a proper behavior of the remote server. The following commands are recognized by `ftp` itself. Command names can be abbreviated to the shortest unique string with identical beginning.

`! [`*command* `[`*args*`]]`
Invoke an interactive shell on the local machine. If there are arguments, the first is taken to be a command to execute directly, with the rest of the arguments as its arguments.

`$` *macro-name* `[`*args*`]`
Execute the macro *macro-name* that was defined with the macdef command. Arguments are passed to the macro unglobbed.

`account [`*passwd*`]`
Supply a supplemental password required by a remote system for access to resources, once a login has been successfully completed. If no argument is included, the user will be prompted for an account password in non-echoing input mode.

`append` *`local-file`* `[`*`remote-file`*`]`
: Append a local file to a file on the remote machine. If *remote-file* is left unspecified, the local file name is used in naming the remote file after being altered by any `ntrans` or `nmap` setting. File transfer uses the current settings for type, format, mode, and structure.

`ascii`
: Set the file transfer type to network ASCII. This is the default type, except when two unices are communicating.

`bell`
: Arrange that a bell be sounded after each file transfer command is completed.

`binary`
: Set the file transfer type to support binary image transfer. This transfer type is selected during initial handshake, should the client on a Unix system recognize that the server is also running on a Unix system.

`bye`
`quit`
: Terminate the FTP session with the remote server and exit `ftp`. An end of file will also terminate the session and exit.

`case`
: Toggle the remote computer's use of letter case mapping during `mget` commands. When `case` is '`on`', a file name at the remote site whose every letter appear in upper case, will be renamed in such a way that all letters are changed to lower case for a local copy of the same file. The default setting is '`off`',

`cd` *`remote-directory`*
: Change the working directory on the remote machine to *remote-directory*.

`cdup`
: Change the remote machine's working directory to the parent of the current working directory.

`chmod` *`mode`* *`file-name`*
: Change the access permission of the file *file-name* on the remote system to *mode*.

`close`
`disconnect`
: Terminate the FTP session with the present remote server, and return to the command interpreter. Any defined macros are erased.

`cr`
: Toggle carriage return stripping during ASCII type file retrieval. Records are denoted by a carriage return/linefeed sequence during ASCII type file transfer. When `cr` is '`on`' (the default), carriage returns are stripped from this sequence to conform with the UNIX single linefeed record delimiter. Records on non-UNIX remote systems may contain single linefeeds; when an ASCII type transfer is made, these linefeeds may be distinguished from a record delimiter only when `cr` is '`off`'.

`delete` *`remote-file`*
: Delete the file *remote-file* on the remote machine.

`debug` `[`*`debug-value`*`]`
: Toggle debugging mode. If an optional *debug-value* is specified it is used to set the debugging level. When debugging is on, `ftp` prints each command sent to the remote machine, preceded by the string '`-->`'.

`dir [`*`remote-directory`*`] [`*`local-file`*`]`

Print a listing of the contents in the directory *remote-directory*, and, optionally, place the output in *local-file*. If interactive prompting is set, `ftp` will prompt the user to verify that the last argument is the intended local file to receive output. If no directory is specified, the current working directory on the remote machine is used. If no local file is specified, or if *local-file* is a dash '`-`', then output is displayed on the terminal.

`epsv4` Toggle the use of EPSV/EPRT for IPv4 addressing. Default is off.

`form` *`format`*

Set the file transfer form to *format*. The only supported format is '`non-print`'.

`get` *`remote-file`* `[`*`local-file`*`]`
`recv` *`remote-file`* `[`*`local-file`*`]`

Retrieve the *remote-file* and store it on the local machine. If a local file name is not specified, the local copy is given the same name as is stated for the remote original, subject to alteration by the current `case`, `ntrans`, and `nmap` settings. The current settings for `type`, `form`, `mode`, and `structure` are effective during file transfer.

`glob` Toggle file name expansion for `mdelete`, `mget`, and `mput`. If globbing is turned off with `glob`, the file name arguments are taken literally and are not expanded. Globbing for `mput` is done as in `csh` syntax. For `mdelete` and `mget`, each remote file name is expanded separately on the remote machine and the lists are not merged. Expansion of a directory name is likely to be different from expansion of the name of an ordinary file: the exact result depends on the remote operating system and on the FTP server, and can be previewed by issuing '`mls remote-files -`'.

Note: `mget` and `mput` are not meant to transfer entire directory subtrees of files. That can be achieved by transferring an already created `tar` or `cpio` archive of the subtree, then making certain that `ftp` uses binary mode.

`hash [`*`size`*`]`

In the absence of an argument, toggle the state of hash-sign ('`#`') printing after each transferred data block. The optional argument selects the size of data blocks, and unconditionally activates printing. The default size is 1024 bytes. For convenience, the size can be written with postfix multipliers 'k', 'K', 'm', 'M', and 'g', 'G', to specify kilobytes, Megabytes, and Gigabytes, respectively.

`help [`*`command`*`]`
`? [`*`command`*`]`

Print an informative message about the meaning of command. If no argument is given, `ftp` prints a list of the known commands.

`idle [`*`seconds`*`]`

Set the inactivity timer on the remote server to *seconds* seconds. If seconds is omitted, the current inactivity timer is printed.

`ipv4` Select IPv4 as the only addressing scheme.

`ipv6` Select IPv6 as the only addressing scheme.

`ipany` Allow IPv4 as well as IPv6 addressing.

`lcd [`*`directory`*`]`
: Change the working directory on the local machine. If no directory is specified, the user's home directory is used.

`lpwd` Print the name of the current working directory on the local machine.

`ls [`*`remote-directory`*`] [`*`local-file`*`]`
: Print a listing of the contents of a directory on the remote machine. The listing includes any system-dependent information that the server chooses to include; for example, most UNIX systems will produce output like the command `ls -l` does. Use `nlist` for a simple file listing.

If *remote-directory* is left unspecified, the current working directory is used. With interactive prompting set, `ftp` will prompt the user to verify that the last argument is indeed the intended local file for storing output. Should no local file be specified, or if *local-file* is a dash '`-`', then output is sent to the terminal.

`macdef` *`macro-name`*
: Define a macro called *macro-name*, with subsequent lines as the macro definition. A null line (consecutive newline characters in a file, or carriage returns at a terminal) terminates macro input mode. There is a limit of 16 macros and a total of 4096 characters shared by all defined macros. Only the first eight characters in *macro-name* are significant when determining which macro to execute. Macros remain defined until a close command is executed.

The macro processor interprets '`$`' and '`\`' as special characters. A '`$`' followed by a number (one or more digits) is replaced by the corresponding argument on the macro's invocation command line. A '`$`' followed by the letter '`i`' tells the macro processor that the macro is to perform a loop. On the first pass, '`$i`' is replaced by the first argument on the macro's invocation command line, while on the second pass it is replaced by the second argument, and so forth. Iteration proceeds until all arguments have been consumed.

A backslash '`\`' followed by any character is replaced by that character. Use the backslash '`\`' to prevent special treatment of the dollar sign '`$`', as was just explained.

`mdelete [`*`remote-files`*`]`
: Delete all *remote-files* on the remote machine.

`mdir` *`remote-files local-file`*
: Like `dir`, except multiple remote files may be specified. If interactive prompting is on, `ftp` will prompt the user to verify that the last argument is indeed the intended local file for storing any output from `mdir`.

`mget` *`remote-files`*
: Expand the *remote-files* on the remote machine and execute a `get` for each file name thus produced. Resulting file names will then be processed according to `case`, `ntrans`, and `nmap` settings. Files are transferred to the local working directory, which can be changed with `lcd directory`; new local directories can be created with `! mkdir directory`.

`mkdir` *`directory-name`*

Make a directory on the remote machine.

`mls` *`remote-files local-file`*

Like `nlist`, except multiple remote files may be specified, and the *local-file* must be specified. If interactive prompting is on, `ftp` will prompt the user to verify that the last argument is the intended local file for storing output. A dash '`-`' is accepted as last argument without check!

`mode [`*`mode-name`*`]`

Set the file transfer mode to *mode-name*. The default mode is '`stream`', and it is also the only implemented mode.

`modtime` *`file-name`*

Show the last modification time of the file on the remote machine.

`mput` *`local-files`*

Consider the arguments to be local names and expand any wild card. Execute a `put` for each file in the resulting list. The remote file names are then computed by use of `ntrans` and `nmap` settings.

`newer` *`file-name`*

Get the file only if the modification time of the remote file is more recent than the file on the current system. If the file does not exist on the current system, the remote file is considered newer. In other respects, this command is identical to `get`.

`nlist [`*`remote-directory`*`] [`*`local-file`*`]`

Print a list of the files in a directory on the remote machine. If *remote-directory* is left unspecified, the current working directory is used. If interactive prompting is on, `ftp` will prompt the user to verify that the last argument is the intended local file for storing output. If no local file is specified, or if *local-file* is '`-`', the output is sent to the terminal.

`nmap [`*`inpattern outpattern`*`]`

Set or unset the file name mapping mechanism. If no arguments are specified, the file name mapping mechanism is unset. Name mapping is applied during `mput` and `put` commands issued without a specified remote target filename. It as also applied to local file names during `mget` and `get` commands issued without local target file name. This command is useful when connecting to a non-UNIX remote computer with different file naming conventions or practices.

The mapping follows the pattern set by *inpattern* and *outpattern*. The template *inpattern* is used on incoming filenames (which may have already been processed according to the `ntrans` and `case` settings). Variable templating is accomplished by including the sequences '`$1`', '`$2`', . . . , '`$9`' in *inpattern*. Use '`\`' to prevent this special treatment of the character '`$`'. All other characters are treated literally, and must be matched in a file name for *inpattern* to bind substrings to variables.

For example, take a pattern '`$1.$2`' and a file name `mydata.data`. Then '`$1`' would have the value '`mydata`', and '`$2`' would be '`data`'.

outpattern determines the final file name. The sequences '`$1`' to '`$9`' are replaced by any values bound to them by *inpattern*. A special sequence '`$0`' always contains the original filename. In addition, a bracketted sequence '`[`*seq1*`,`*seq2*`]`' expands to *seq1* if *seq1* contains a non-empty string, and expands to *seq2* otherwise. For example, the command

```
nmap $1.$2.$3 [$1,$2].[$2,file]
```

would yield the output file name `myfile.data` for input names `myfile.data` and `myfile.data.old`, but produces `myfile.file` from the input `myfile`, and `myfile.myfile` from `.myfile`.

Spaces may be included in *outpattern*, but are easily removed:

```
nmap $1 |sed "s/ *$//" > $1
```

Use a backslash '\' to escape the characters '`$`', '`[`', '`]`', and '`,`'.

`ntrans [`*inchars* `[`*outchars*`]]`

Set or unset the filename character translation mechanism. If no arguments are specified, the filename character translation mechanism is unset. If arguments are specified, characters in remote filenames are translated during `mput` commands and `put` commands issued without a specified remote target filename. If arguments are specified, characters in local filenames are translated during `mget` commands and `get` commands issued without a specified local target filename. This command is useful when connecting to a non-UNIX remote computer with different file naming conventions or practices.

Characters in a filename matching a character in *inchars* are replaced with the corresponding character in *outchars*. If the character's position in *inchars* is longer than the length of *outchars*, the character is deleted from the file name.

`open` *host* `[`*port*`]`

Establish a connection to the specified FTP server at *host*. An optional port number may be supplied, in which case, `ftp` will attempt to contact the server at that specific TCP port. If the `autologin` option is on (is so by default), `ftp` will also attempt to automatically log the user in to the FTP server.

`passive` Toggle passive mode. If passive mode is turned on (default is off), the `ftp` client will send a `PASV` command for all data connections instead of the usual `PORT` command. The `PASV` command requests that the remote server open a port for the data connection and return the address of that port. The remote server listens on that port and the client connects to it. When using the more traditional `PORT` command, the client listens on a port and sends that address to the remote server, who connects back to it. Passive mode is useful when using `ftp` through a gateway router or host that controls the directionality of traffic. (Note that though `ftp` servers are required to support the `PASV` command by RFC 1123, some do not.) If `epsv4` has been set to on, the client will attempt `EPSV` before `PASV` for IPv4. As a last resort `LPSV` is attempted. With IPv6 only `EPSV` and `LPSV` are possible.

`prompt` Toggle interactive prompting. Interactive prompting occurs during multiple file transfers to allow the user to selectively retrieve or store files. If prompting is turned off (default is on), any `mget` or `mput` will transfer all files, and any `mdelete` will delete all files.

`proxy` *`ftp-command`*

Execute an `ftp` command on a secondary control connection. This command allows simultaneous connection to two remote FTP servers for transferring files between the two servers. The first proxy command should be `open`, to establish the secondary control connection. Enter the command `proxy ?` to see other commands usable for the secondary connection. The following commands behave differently when prefaced by `proxy`: `open` will not define new macros during the auto-login process, `close` will not erase existing macro definitions, `get` and `mget` transfer files from the host on the primary control connection to the host on the secondary control connection, and `put`, `mput`, and `append` transfer files from the host on the secondary control connection to the host on the primary control connection.

Note that the protocol command `PASV` must be understood by the server on the secondary control connection for this kind of file transfer to succeed.

`put` *`local-file`* `[`*`remote-file`*`]`
`send` *`local-file`* `[`*`remote-file`*`]`

Store a local file on the remote machine. If *remote-file* is left unspecified, the local file name is used after processing according to any `ntrans` or `nmap` settings in naming the remote file. File transfer uses the current settings for type, format, mode, and structure.

`pwd` Print the name of the current working directory on the remote machine.

`quote` *`arg`*`...`

The arguments specified are sent, verbatim, to the remote FTP server.

`reget` *`remote-file`* `[`*`local-file`*`]`

`reget` acts like `get`, except that if *local-file* exists and is smaller than *remote-file*, then *local-file* is presumed to be a partially transferred copy of *remote-file* and the transfer is continued from the apparent point of failure. This command is useful when transferring very large files over networks that are prone to dropping connections.

`rhelp` `[`*`command-name`*`]`

Request help from the remote FTP server. If *command-name* is specified it is passed to the server as well.

`rstatus` `[`*`file-name`*`]`

With no arguments, show status of remote machine. If filename is specified, show status of *file-name* on remote machine.

`rename` `[`*`from`*`]` `[`*`to`*`]`

Rename the file *from* on the remote machine as *to*. Name mapping takes effect without *to*.

`reset` Clear reply queue. This command re-synchronizes command/reply sequencing with the remote FTP server. Resynchronization may be necessary following a violation of the FTP protocol by the remote server.

`restart` *marker*
: Restart the immediately following `get` or `put` at the indicated marker. On UNIX systems, `marker` is usually a byte offset into the file.

`rmdir` *directory-name*
: Delete a directory on the remote machine.

`runique`
: Toggle the storing of files on the local system with unique filenames. If a file already exists with a name equal to the inteded local file name for a `get` or `mget` command, then a string '`.1`' is appended to the name. If the resulting name matches another existing file, '`.2`' is appended to the original name. If this process continues up to '`.99`', an error message is printed, and the transfer does not take place. The generated unique filename will be reported. Note that `runique` will not affect local files generated from a shell command. The default value is off.

`sendport`
: Toggle the use of `PORT` commands. By default, `ftp` will attempt to use a `PORT` command when establishing a connection for each data transfer. The use of `PORT` commands can prevent delays when performing multiple file transfers. If the `PORT` command fails, `ftp` will use the default data port. When the use of `PORT` commands is disabled, no attempt will be made to use `PORT` commands for each data transfer. This is useful for certain FTP implementations which do ignore `PORT` commands but, incorrectly, indicate they've been accepted.

`site` *arg*`...`
: The arguments specified are sent, verbatim, to the remote FTP server as a `SITE` command.

`size` *file-name*
: Return size of *file-name* on remote machine.

`status`
: Show the current status of `ftp`.

`struct [`*struct-name*`]`
: Set the file transfer structure to *struct-name*. By default '`file`' structure is used, which also is the only supported value.

`sunique`
: Toggle storing of files on remote machine under unique file names. Remote FTP server must support FTP protocol `STOU` command for successful completion. The remote server will report unique name. Default value is off.

`system`
: Show the type of operating system running on the remote machine.

`tenex`
: Set the file transfer type to that needed to talk to TENEX machines.

`trace`
: Toggle packet tracing (feature is not implemented).

`type [`*type-name*`]`
: Set the file transfer type to *type-name*. If no type is specified, the current type is printed. The recognized type names are '`ascii`', '`binary`', '`ebcdic`', '`image`', and '`tenex`'. The default type is network ASCII.

`umask [`*newmask*`]`
: Set the default umask on the remote server to *newmask*. If *newmask* is omitted, the current umask is printed.

`user` *user-name* [*password*] [*account*]
: Identify yourself to the remote FTP server. If the password is not specified and the server requires it, `ftp` will prompt the user for it (after disabling local echo). If an account field is not specified, and the FTP server requires it, the user will be prompted for it. If an account field is specified, an account command will be relayed to the remote server after the login sequence is completed if the remote server did not require it for logging in. Unless `ftp` is invoked with `auto-login` disabled, this process is done automatically on initial connection to the FTP server.

`verbose` Toggle verbose mode. In verbose mode, all responses from the FTP server are displayed to the user. In addition, if verbose is on, when a file transfer completes, statistics regarding the efficiency of the transfer are reported. By default, verbose is on.

Command arguments which have embedded spaces may be inclosed within citation characters '`"`'.

10.3 Environment variables in use

`ftp` accesses the following environment variables.

`HOME` Used for locating a `.netrc` file, if one exists.

`NETRC` Alternate location of the `.netrc` file, taking precedence over the standard location.

`SHELL` For determining the default shell interpreter.

10.4 Aborting a file transfer

To abort a file transfer, use the terminal interrupt key (usually *C-c*). Sending transfers will be immediately halted. Receiving transfers will be halted by sending a FTP protocol command `ABOR` to the remote server, discarding any further data received. The speed at which this is accomplished depends upon the remote server's support for `ABOR` processing. If the remote server does not support the `ABOR` command, an '`ftp>`' prompt will not appear until the remote server has completed sending the requested file.

The terminal interrupt key sequence will be ignored when `ftp` has completed any local processing and is awaiting a reply from the remote server. A long delay in this mode may result from the `ABOR` processing described above, or from unexpected behavior by the remote server, including violations of the FTP protocol. If the delay results from unexpected remote server behavior, the local `ftp` program must be killed by hand.

10.5 File naming conventions

Files specified as arguments to `ftp` commands are processed according to the following rules.

1. If the file name '`-`' is specified, standard input (for reading) or standard output (for writing) is used.

2. If the first character of the file name is '`|`', the remainder of the argument is interpreted as a shell command. `ftp` then forks a shell, using `popen` with the argument supplied, and reads/writes from standard input/output. If the shell command includes spaces, the argument must be quoted; e.g. '`"ls -lt"`'.

   A particularly useful example of this mechanism in action, is

   ```
   ftp> dir . |less
   ```

   which allows the user to scroll through a long directory listing.
3. Failing the above checks, if *globbing* is enabled, local file names are expanded according to the rules used by `csh`; c.f. the `glob` command. If the `ftp` command expects a single local file (e.g. `put`), only the first filename generated by the globbing operation is used.
4. For the commands `mget` and `get` with unspecified local file name, the local file name is set to the remote file name, which may be altered by a `case`, `ntrans`, or `nmap` settings. The resulting file name may then be modified if `runique` is set.
5. For the commands `mput` and `put` with unspecified remote file name, the remote file name is copied from the local file name, which may be altered by a `ntrans` or `nmap` settings. The resulting file name may also be modified by the remote server if `sunique` is set.

10.6 File transfer parameters

The FTP specification includes many parameters which may affect a file transfer. The type may be one of '`ascii`', '`image`' (binary), '`ebcdic`', and '`local`' byte size (for PDP-10's and PDP-20's mostly). `ftp` supports the '`ascii`' and '`image`' types of file transfer, plus local byte size 8 for tenex mode transfers.

`ftp` supports only the default values for the remaining file transfer parameters: `mode`, `form`, and `struct`.

An error in the treatment of carriage returns in the 4.2BSD ascii-mode transfer code has been corrected by the present implementation. This correction may result in corrupt transfers of binary files to and from 4.2BSD servers, when done using the ascii type. Avoid this problem by using the binary image type.

10.7 The `.netrc` file

The `.netrc` file contains login and initialization information used by the auto-login process. It generally resides in the user's home directory, but a location outside of the home directory can be set using the environment variable `NETRC`. Both locations are overridden by the command line option `-N`. The selected file must be a regular file, or access will be denied.

The following tokens are recognized; they may be separated by spaces, tabs, or new-lines:

'`machine name`'

Identify a remote machine name. The auto-login process searches the `.netrc` file for a machine token that matches the remote machine specified on the `ftp` command line or as an open command argument. Once a match is made, the subsequent `.netrc` tokens are processed, stopping when the end of file is reached or another machine or a default token is encountered.

'default' This is the same as machine name except that default matches any name. There can be only one default token, and it must be after all machine tokens. This is normally used as:

```
default login anonymous password user@site
```

thereby giving the user automatic anonymous ftp login to machines not specified in .netrc. This can be overridden by using the -n flag to disable auto-login.

'login name'
Identify a user on the remote machine. If this token is present, the auto-login process will initiate a login using the specified name.

'password string'
Supply a password. If this token is present, the auto-login process will supply the specified string if the remote server requires a password as part of the login process. Note that if this token is present in the .netrc file for any user other than anonymous, ftp will abort the auto-login process if the .netrc is readable by anyone besides the user.

'account string'
Supply an additional account password. If this token is present, the auto-login process will supply the specified string if the remote server requires an additional account password, or the auto-login process will initiate an ACCT command if it does not.

'macdef name'
Define a macro. This token functions like the ftp macdef command functions. A macro is defined with the specified name; its contents begin with the next .netrc line and continue until a null line (consecutive new-line characters) is encountered. If a macro named init is defined, it is automatically executed as the last step in the auto-login process.

11 rcp: Copy files between machines

`rcp` copies files between machines. Each file or directory argument is either a remote file name of the form '`rname@rhost:path`', or a local file name (containing no '`:`' characters, or a '`/`' before any '`:`'s).

Synopsis:

```
rcp [option]... old-file new-file
rcp [option]... files... directory
```

11.1 Command line options

`-4`
`--ipv4` Use only IPv4.

`-6`
`--ipv6` Use only IPv6.

`-d` *directory*
`--target-directory=`*directory*
: Copy all source arguments into *directory*.

`-f`
`--from` (Server mode only.) Copying from remote host.

`-k` *realm*
`--realm=`*realm*
: The option requests rcp to obtain tickets for the remote host in realm *realm* instead of the remote host's realm.

`-K`
`--kerberos`
: Turns off all Kerberos authentication.

`-p`
`--preserve`
: Causes `rcp` to attempt to preserve (duplicate) in its copies the modification times and modes of the source files, ignoring the umask. By default, the mode and owner of the target file are preserved if the target itself already exists; otherwise the mode of the source file is modified by the `umask` setting on the destination host.

`-r`
`--recursive`
: If any of the source files are directories, `rcp` copies each subtree rooted at that name; in this case the destination must be a directory.

`-t`
`--to` (Server mode only.) Copying to remote host.

`-x`
`--encrypt`
: Turns on encryption for all data passed via the `rcp` session. This may impact response time and CPU utilization, but provides increased security.

`rcp` doesn't detect all cases where the target of a copy might be a file in cases where only a directory should be legal.

`rcp` can be confused by any output generated by commands in a `.login`, `.profile`, or `.cshrc` file on the remote host.

The destination user and hostname may have to be specified as '`rhost.rname`' when the destination machine is running the 4.2BSD version of `rcp`.

12 rexec: a remote execution program

`rexec` is a program that executes a program on another host.

Synopsis:

```
rexec --user=login --password=pass --host=host  \
      [OPTION] command
```

12.1 Command line options

`-4`
`--ipv4` Use only IPv4 connections as all times.

`-6`
`--ipv6` Use only IPv6 connections.

`-a`
`--ipany` Allow any address family for connections. This is the default.

`-e`
`--error=`*port*
Specify the TCP port to use for stderr redirection, in case it is not specified a random port will be used.

`-h`

`--host=`*name*
Specify the host with whom to connect: symbolic name or address.

`-n`
`--noerr` If specified, an error stream will not be created.

`-p`
`--password=`*passwd*
Specify the password for logging-in. The special value consisting of a single dash '`-`' will make `rexec` read a single line from stdin. This input is then used as password and is passed as such to the remote server. Thus it is possible to hide vital access information slightly better than the full disclosure implicit in the text of a command line option.

`-P`
`--port=`*num*
Specify to which numerical port a connection shall be sought. If it is not specified, then use port 512/tcp by default.

`-u`
`--user=`*name*
Specify the user with whom to log into the server.

13 rlogin: Remote login

The rlogin command starts a terminal session on the specified remote host, provided the required authentication is successful. The remote terminal type is the same as that given in the TERM local environment variable. The terminal and the window size stay the same, if the remote host supports them, and any changes in size are transferred as need may be.

When using the rlogin command, you can create a link in your path, using a host name as the link name. For example:

```
# ln -s /usr/bin/rlogin hostname
# hostname -8
```

Afterwards, the use of *hostname* will automatically invoke rlogin to direct a log in request to the remote host named *hostname*.

rlogin allows access to the remote host without the use of a password. The prerequisite is a suitable specification in ~/.rhosts. For details, See Section "rcmd" in *The GNU C Library Reference Manual*.

13.1 Command line options

The options are as follows :

-4
--ipv4 Use only IPv4.

-6
--ipv6 Use only IPv6.

-8
--8-bit Allows an eight-bit input data path at all times; otherwise parity bits are stripped except when the remote side's stop and start characters are other than *C-S*/*C-Q*.

-d
--debug Turns on socket debugging on the TCP sockets used for communication with the remote host.

-e *char*
--escape=*char*
Allows user specification of the escape character, which is '~' by default. This specification may be as a literal character, or as an octal value in the form '\nnn'.

-E
--no-escape
Stops any character from being recognized as an escape character. When used with the -8 option, this provides a completely transparent connection.

-l *user*
--user=*user*
By default, the remote username is the same as the local username. This option, and the 'user@host' format, allow the remote user name to be made explicit, or changed.

The next three options are available only if the program has been compiled with support for Kerberos authentication.

`-k` *`realm`*
`--realm=`*`realm`*

The option requests rlogin to obtain tickets for the remote host in realm *realm* instead of the remote host's realm.

`-K`
`--kerberos`

Turns off all Kerberos authentication.

`-x`
`--encrypt`

Turns on encryption for all data passed via the rlogin session. This may impact response time and CPU utilization, but provides increased security.

13.2 Escape characters and flow control

As long as the connection stands, the client program `rsh` is observing the input stream in order to detect so called escape sequences, allowing the user to execute some local actions without having to tear down the remote connection.

The sequences consist of two characters, the first of which always is the distinguished character *escape-char*. The following sequences are supported:

- *`escape-char`* `.` disconnects from the remote host.
- *`escape-char`* *`C-d`* does the same. (Termios character '`VEOF`'.)
- *`escape-char`* *`C-z`* suspends the session, halting the remote process, but keeping it ready for resumed processing. The user is given access to a local shell. (Termios character '`VSUSP`'.)
- *`escape-char`* *`delayed-suspend-char`* implements a half-way suspend, in the sense of stopping local input from reaching the remote side, but still displaying all output from the remote host on the local terminal. The remote process is still running, but the local user is given a local shell until the resuming the original command. Normally, only BSD systems offer this mode. (Termios character '`VDSUSP`'.)

By default, the character tilde '`~`' is assigned to *escape-char*, but it can be changed using the option `--escape`. The processing of escape sequences can even be disable using the option `--no-escape`. On BSD systems, *delayed-suspend-char* is usually set to *`C-Y`*. It displays as '`dsusp`' using `stty`.

All echoing takes place at the remote site, so that the `rlogin` is transparent except possibly for transmission delays. Flow control via *`C-S`* and *`C-Q`*, if at all supported, will stop and start the flow of data on the local terminal. Flushing of input and output on interrupts is also handled properly.

On the server side the `iruserok` and `ruserok` functions are used to authenticate the connection request, unless Kerberised mode is in effect. See the appropriate man pages for more information.

13.3 Kerberos Authentication

If `rlogin` was compiled with kerberos support, options `-x`, `-k`, `-K` are available. Each user may have a private authorization list in the file `.k5login` in their home directory. Each line in this file should contain a Kerberos principal name of the form '`principal/instance@realm`'. If the originating user is authenticated to one of the principals named in `.k5login`, access is granted to the account. The principal '`accountname@localrealm`' is granted access if there is no `.k5login` file. Otherwise a login and password will be prompted for on the remote machine as in `login`. To avoid certain security problems, the `.k5login` file must be owned by the remote user. If Kerberos authentication fails, a warning message is printed and the standard Berkeley rlogin is used instead.

14 rsh: Remote shell

rsh executes commands on a remote host and copies its local standard input to that of the remote command, as well as the remote standard output to the local standard output, and the remote standard error to the local standard error. Locally raised interrupt, quit and terminate signals are all propagated to the remote command. Normally rsh terminates when the remote command does so.

When using the rsh command, you can for convenience create a link in your path, using a host name as name of the link. For example:

```
# ln -s /usr/bin/rsh hostname
# hostname ls
```

Afterwards, *hostname* will be passed to rsh as host name whenever the command *hostname* is issued.

rsh allows access to the remote host without the use of a password. The prerequisite is a suitable specification in ~/.rhosts. For details, See Section "rcmd" in *The GNU C Library Reference Manual*.

If no command is specified for rsh ar argument following the host name, then you will be logged in on the remote host using rlogin.

14.1 Command line options

The options are as follows :

-4
--ipv4 Use only IPv4.

-6
--ipv6 Use only IPv6.

-d
--debug Turns on socket debugging used for communication with the remote host.

-l *user*
--user=*user*
By default, the remote username is the same as the local username. The -l option and the 'username@host' format allow the remote user name to be specified. Kerberos authentication is used, whenever available, and authorization is determined as in rlogin (see Chapter 13 [rlogin invocation], page 32).

-n
--no-input
Use /dev/null for all input, telling the server side that we send no material. This can prevent the remote process from blocking, should it optionally accept more input. The option is void together with encryption.

The next three options are available only if the program has been compiled with support for Kerberos authentication.

`-k` *realm*
`--realm=`*realm*

The option requests rsh to obtain tickets for the remote host in realm *realm* instead of the remote host's realm.

`-K`
`--kerberos`

Turns off all Kerberos authentication.

`-x`
`--encrypt`

Turns on encryption for all data passed via the rsh session. This may impact response time and CPU utilization, but provides increased security.

Finally, some compatibility options are present:

`-8`
`--8-bit`
`-e` *char*
`--escape=`*char*
`-E`
`--no-escape`

Ignored during normal operation, but passed on to `rlogin` when `rsh` is invoked without a command argument.

14.2 Note on stream redirections

Beware that non-quoted shell metacharacters are interpreted on the local machine, while quoted metacharacters are interpreted on the remote machine. For example:

```
rsh otherhost  cat remotefile >> localfile
rsh otherhost  cat remotefile ">>" otherfile
```

The first command appends the contents of `remotefile`, as found on the remote host, to the file `localfile` on the local host, since the local shell will intercept the redirection and will thus receive whatever the remote process directs to stdout.

In contrast, the second command will append the contents of the same file `remotefile` to a file named `otherfile` again, but this time the file is located on the remote host. The effect of quoting the redirection operator is to execute the command

```
cat remotefile >> localfile
```

entirely on the remote most, whence stdout at the remote host will have nothing to transmit to the listening local host!.

15 talk: a communication program

talk is a visual communication program which copies lines from your terminal to that of another user.

Synopsis:

```
talk person [ttyname]
```

15.1 Invoking

The command line arguments are as follows:

person If you wish to talk to someone on your own machine, then *person* is just the other person's login name. If you wish to talk to a user on another host, then *person* is of the form 'user@host'.

ttyname If you wish to talk to a local user who is logged in more than once, the argument *ttyname* may be used to indicate the appropriate terminal name, where *ttyname* typically is of the form 'ttyXX', or 'pts/X'.

When first called, talk sends a message to the addressed user:

```
Message from TalkDaemon@his_machine...
talk: connection requested by your_name@your_machine.
talk: respond with: talk your_name@your_machine
```

At this point, the recipient of the message could elect to accept the call and to establish a connection by typing:

```
talk your_name@your_machine
```

It doesn't matter from which machine the recipient replies, as long as his login-name is the same. Once communication is established, the two parties may type text simultaneously, with their output appearing in separate windows. Typing *C-L* will cause the screen to be reprinted, while erase, kill, and word kill characters will behave normally. In addition, *C-D* will cause both windows to be locally cleared of all text. This keystroke will appear as a simple '^D' on the remote terminal, though. It signals to the other party that you yourself have just cleared your terminal of all text.

To exit, just type an interrupt character *C-C*; talk then moves the cursor to the bottom of the screen and restores the terminal to its previous state.

The ability to talk may be enabled or disabled by use of the mesg command. It is system dependent whether this message passing is enabled at the outset of a terminal session. Certain commands, in particular nroff and pr, disable messages in order to prevent messy output.

16 telnet: User interface to TELNET

Login to a remote system HOST, optionally using a (non-standard) service port PORT.

Synopsis:

```
telnet [OPTION...] [HOST [PORT]]
```

16.1 Command line options

`-4`
`--ipv4` Use only IPv4.

`-6`
`--ipv6` Use only IPv6.

`-8`
`--binary` Use an 8-bit data path.

`-a`
`--login` Attempt automatic login.

`-c`
`--no-rc` Do not read the user's file `$HOME/.telnetrc`.

`-d`
`--debug` Turn on socket level debugging.

`-e` *char*
`--escape=`*char*
Use *char* as escape character.

`-E`
`--no-escape`
Do not use an escape character.

`-k` *realm*
`--realm=`*realm*
Request Kerberos realm *realm* instead of whatever is declared as default realm in the system's or user's settings.

`-K`
`--no-login`
Do not automatically login to the remote system.

`-l` *user*
`--user=`*user*
Attempt automatic login as *user*.

`-L`
`--binary-output`
Use an 8-bit data path for output only.

`-n` *file*
`--trace=`*file*
Record trace information into *file*.

`-r`
`--rlogin` Display a user-interface similar to that of `rlogin`.

`-x`
`--encrypt`
If possible, encrypt the data stream.

`-X` *atype*
`--disable-auth=`*atype*
Disable authentication of type *atype*. Use this option multiple times if more than one type is to be disabled. Standard choices are ‘`null`’, ‘`kerberos_v4`’, and ‘`kerberos_v5`’.

17 tftp: TFTP client

`tftp` is the user interface to the Internet TFTP, Trivial File Transfer Protocol, which allows users to transfer files to and from a remote machine. The remote host may be specified on the command line, in which case `tftp` uses host as the default host for future transfers.

Synopsis:

```
tftp [option]... host
```

17.1 Commands

Once `tftp` is running, it issues the prompt and recognizes the following commands:

`? `*`command-name`*
: Print help information.

`ascii` Shorthand for `mode ascii`

`binary` Shorthand for `mode binary`

`connect `*`host-name`*` [`*`port`*`]`
: Set the host (and optionally port) for transfers. Note that the TFTP protocol, unlike the FTP protocol, does not maintain connections between transfers; thus, the connect command does not actually create a connection, but merely remembers what host is to be used for transfers. You do not have to use the connect command; the remote host can be specified as part of the get or put commands.

`get `*`file-name`*
`get `*`remotename localname`*
`get `*`file...`*
: Get a file, or a set of files, from the specified sources. The source can be in one of two forms: a file name on the remote host, if the host has already been specified, or a string of the form '`host:filename`' to specify both a host and file name at the same time. If the latter form is used, the last hostname specified becomes the default for future transfers. When specifying a numeric IPv6 address as host part, then this address must be enclosed between square brackets, since it contains colons and would interfere with the delimiter before the file name. Brackets are optional for IPv4 addresses.

```
tftp> get [2001:1234::12]:issue
```

`mode `*`transfer-mode`*
: Set the mode for transfers; *transfer-mode* may be one of '`ascii`' or '`binary`'. The default is '`ascii`'.

`put `*`file`*
`put `*`localfile remotefile`*
`put `*`file... remote-directory`*
: Put a file or set of files to the specified remote file or directory. The destination can be in one of two forms: a filename on the remote host, if the host has already been specified, or a string of the form '`host:filename`' to specify both

a host and filename at the same time. If the latter form is used, the hostname specified becomes the default for future transfers. If the `remote-directory` form is used, the remote host is assumed to be a UNIX machine. The same use of square brackets for enclosing numeric IPv6 addresses applies here, as was mentioned for the command `get`.

`quit` Exit `tftp`. An end of file also exits.

`rexmt` *`retransmission-timeout`*
Set the per-packet retransmission timeout, in seconds.

`status` Show current status.

`timeout` *`total-transmission-timeout`*
Set the total transmission timeout, in seconds.

`trace` Toggle packet tracing.

`verbose` Toggle verbose mode.

Because there is no user-login or validation within the `tftp` protocol, the remote site will probably have some sort of file-access restrictions in place. The exact methods are specific to each site and therefore difficult to document here.

18 `inetd`: Internet super-server

`inetd` program should be run at boot time by /etc/rc. It then listens for connections on certain internet sockets. When a connection is found on one of its sockets, it decides what service the socket corresponds to, and invokes a program to service the request. The server program is invoked with the service socket as its standard input, output and error descriptors. After the program is finished, inetd continues to listen on the socket (except in some cases which will be described below). Essentially, inetd allows running one daemon to invoke several others, reducing load on the system.

There are two types of services that inetd can start: standard and TCPMUX. A standard service has a well-known port assigned to it; it may be a service that implements an official Internet standard or is a BSD-specific service. As described in RFC 1078, TCPMUX services are nonstandard services that do not have a well-known port assigned to them. They are invoked from inetd when a program connects to the "tcpmux" well-known port and specifies the service name. This feature is useful for adding locally-developed servers.

18.1 Invocation

Normally, `inetd` is invoked without any arguments. It does, however, support several command line options. These are:

`-d`
`--debug` Turns on debugging. With this option, `inetd` stays in foreground and prints additional debugging information of stderr.

`--environment`
Pass local and remote socket information in environment variables. See Section 18.5 [Inetd Environment], page 45.

`-p[`*`file`*`]`
`--pidfile[=`*`file`*`]`
Use *file* as location to store process ID of the running server process, thus overriding the default location. Setting an empty argument will disable the use of a file for storing the process ID.

`--resolve`
Resolve IP addresses when setting environment variables. See Section 18.5 [Inetd Environment], page 45.

`-R` *`rate`*
`--rate=`*`rate`*
Specify the maximum number of times a service can be invoked in one minute; the default is 1000.

18.2 Configuration file

Upon execution, inetd reads its configuration information from configuration files and directories named on the command line. By default these are `/etc/inetd.conf` and `/etc/initd.d`. If the configuration pathname is a directory, all files in the directory are read and interpreted like a configuration file. All of the configuration files are read and the results are merged.

There must be an entry for each field in the configuration file, with entries for each field separated by a tab or a space. Comments are denoted by a "#" at the beginning of a line. The available fields of the configuration file are summarized in the table below (optional parts are enclosed in square brackets):

[service node:]service name

The service-name entry is the name of a valid service in the file `/etc/services`. For "internal" services (see Section 18.3 [Built-in services], page 44), the service name must be the official name of the service (that is, the first entry in `/etc/services`), or a numeric representation thereof. For TCPMUX services, the value of the '`service name`' field consists of the string '`tcpmux`' followed by a slash and the locally-chosen service name (see Section 18.4 [TCPMUX], page 45).

An optional '`service node`' prefix is allowed for internet services. When present, it supplies the local addresses `inetd` should use when listening for that service. '`Service node`' consists of a comma-separated list of addresses. Both symbolic host names and numeric IP addresses are allowed. Symbolic hostnames are looked up in DNS service. If a hostname has multiple address mappings, `inetd` creates a socket to listen on each address.

To avoid repeating an address that occurs frequently, a line with a host address specifier and colon, but no further fields is allowed, e.g.:

```
127.0.0.1,192.168.0.5:
```

The address specifier from such a line is remembered and used for all further lines lacking an explicit host specifier. Such a default address remains in effect until another such line or end of the configuration is encountered, whichever occurs first.

A special hostname '`*`' stands for the wildcard address. When used in a normal configuration line, it causes the default address specifier to be ignored for that line. When used in a default address specification, e.g.:

```
*:
```

it causes any previous default address specifier to be forgotten.

socket type

The socket type should be one of '`stream`', '`dgram`', '`raw`', '`rdm`', or '`seqpacket`', depending on whether the socket is a stream, datagram, raw, reliably delivered message, or sequenced packet socket. TCPMUX services must use '`stream`'.

protocol

The protocol must be a valid protocol as given in `/etc/protocols`. Examples might be '`tcp`' or '`udp`'. TCPMUX services must use '`tcp`'. If IPv6 support is enabled the sockets will accept both IPv4 and IPv6 connections if that is supported by the OS. If inetd should only accept IPv4 or IPv6 connections, add '`4`' or '`6`' to the protocol name. For example '`tcp4`' will only accept IPv4 tcp connections and '`udp6`' will only accept IPv6 udp connections.

wait/nowait[.max]

The '`wait/nowait`' entry specifies whether the server that is invoked by `inetd` will take over the socket associated with the service access point, and thus whether inetd should wait for the server to exit before listening for new service requests. Datagram servers must use '`wait`', as they are always invoked with the original datagram socket bound to the specified service address. These servers must read at least one datagram from the socket before exiting. If a datagram server connects to its peer, freeing the socket so inetd can received further messages on the socket, it is said to be a "multi-threaded" server; it should read one datagram from the socket and create a new socket connected to the peer. It should fork, and the parent should then exit to allow inetd to check for new service requests to spawn new servers. Datagram servers which process all incoming datagrams on a socket and eventually time out are said to be "single-threaded". `comsat` and `talkd` are both examples of the latter type of datagram server. `tftpd` is an example of a multi-threaded datagram server.

Servers using stream sockets generally are multi-threaded and use the '`nowait`' entry. Connection requests for these services are accepted by inetd, and the server is given only the newly-accepted socket connected to a client of the service. Most stream-based services and all TCPMUX services operate in this manner. For such services, the number of running instances of the server can be limitied by specifying optional '`max`' suffix (a decimal number), e.g.: '`nowait.15`'.

Stream-based servers that use '`wait`' are started with the listening service socket, and must accept at least one connection request before exiting. Such a server would normally accept and process incoming connection requests until a timeout. Other services must use '`nowait`'.

user

The user entry should contain the user name of the user as whom the server should run. This allows for servers to be given less permission than root. An optional form includes also a group name as a suffix, separated from the user name by colon or a period, i.e., '`user:group`' or '`user.group`'.

server program

The server-program entry should contain the pathname of the program which is to be executed by inetd when a request is found on its socket. If inetd provides this service internally, this entry should be '`internal`'.

It is common usage to specify `/usr/sbin/tcpd` in this field.

server program arguments

The server program arguments should be just as arguments normally are, starting with `argv[0]`, which is the name of the program. If the service is provided internally, this entry must contain the word '`internal`', or be empty.

18.3 Built-in services

The `inetd` program provides several "trivial" services internally by use of routines within itself. All these services can operate both in '`stream`' and in '`dgram`' mode. They are:

echo
Send back to the originating source any data received from it. This is a debugging and measurement tool.

discard
Silently throw away any data received.

chargen
This is a character generator service. It can be operated as both stream or dgram service. When operating in ‘`stream`’ mode, once a connection is established a stream of data is sent out the connection (and any data received is thrown away). This continues until the calling user terminates the connection. When operating in ‘`dgram`’ mode, `inetd` listens for UDP datagrams, and for each received datagram, answers with a datagram containing a random number (between 0 and 512) of characters. Any data in the received datagram are ignored.

daytime
Send back the current date and time in a human readable form. Any input is discarded.

time
Send back the current date and time as a 32-bit integer number, nrepresenting the number of seconds since midnight, January 1, 1900.

18.4 TCPMUX

The TCPMUX protocol.

> A TCP client connects to a foreign host on TCP port 1. It sends the service name followed by a carriage-return line-feed <CRLF>. The service name is never case sensitive. The server replies with a single character indicating positive (+) or negative (-) acknowledgment, immediately followed by an optional message of explanation, terminated with a <CRLF>. If the reply was positive, the selected protocol begins; otherwise the connection is closed.” The program is passed the TCP connection as file descriptors 0 and 1.

If the TCPMUX service name begins with a “`+`”, `inetd` returns the positive reply for the program. This allows you to invoke programs that use stdin/stdout without putting any special server code in them.

The special service name ‘`help`’ causes inetd to list TCPMUX services in `inetd.conf`.

To define TCPMUX services, the configuration file must contain a ‘`tcpmux internal`’ definition.

Here are several example service entries for the various types of services:

```
ftp          stream  tcp   nowait root   /usr/libexec/ftpd     ftpd -l
ntalk        dgram   udp   wait   nobody:tty /usr/libexec/talkd talkd
tcpmux       stream  tcp   nowait root   internal
tcpmux/+date stream  tcp   nowait guest /bin/date              date
tcpmux/phonebook stream tcp nowait guest /usr/bin/phonebook    phonebook
```

18.5 Inetd Environment

If a connection is made with a streaming protocol (‘`stream`’) and if `--environment` option has been given, `inetd` will set the following environment variables before starting the program:

PROTO Always 'TCP'.

TCPLOCALIP
: Local IP address of the interface which accepted the connection.

TCPLOCALPORT
: Port number on which the TCP connection was established.

TCPREMOTEIP
: IP address of the remote client.

TCPREMOTEPORT
: Port number on the client side of the TCP connection.

Additionally, if given the `--remote` option, `inetd` sets the following environment variables:

TCPLOCALHOST
: DNS name of `TCPLOCALIP`.

TCPREMOTEHOST
: DNS name of `TCPREMOTEIP`.

18.6 Error Messages

The inetd server logs error messages using syslog. Important error messages and their explanations are:

'`service/protocol server failing (looping), service terminated.`'

The number of requests for the specified service in the past minute exceeded the limit. The limit exists to prevent a broken program or a malicious user from swamping the system. This message may occur for several reasons:

1. there are lots of hosts requesting the service within a short time period,
2. a "broken" client program is requesting the service too frequently,
3. a malicious user is running a program to invoke the service in a "denial of service" attack,
4. the invoked service program has an error that causes clients to retry quickly.

Use the `-R` option, as described above, to change the rate limit. Once the limit is reached, the service will be reenabled automatically in 10 minutes.

'`service/protocol: No such user 'user', service ignored`'
'`service/protocol: getpwnam: user: No such user`'

No entry for user exists in the passwd file. The first message occurs when inetd (re)reads the configuration file. The second message occurs when the service is invoked.

'`service/protocol: No such user 'user', service ignored`'
'`service/protocol: getpwnam: user: No such user`'

No entry for user exists in the passwd file. The first message occurs when inetd (re)reads the configuration file. The second message occurs when the service is invoked.

'`service: can't set uid number`'
'`service: can't set gid number`'
: The user or group ID for the entry's user is invalid.

19 syslogd: system service logging faclity

`syslogd` is a system service that provides error logging facility. Messages are read from the UNIX domain socket `/dev/log`, from an Internet domain socket specified in `/etc/services`, and from the special device `/dev/klog` (to read kernel messages).

`syslogd` creates the file `/var/run/syslog.pid`, and stores its process id there. This can be used to kill or reconfigure `syslogd`.

The message sent to `syslogd` should consist of a single line. The message can contain a priority code, which should be a preceding decimal number in angle braces, for example, `<5>`. This priority code should map into the priorities defined in the include file `sys/syslog.h`.

```
syslogd [options]...
```

`-f file`
`--rcfile=file`
: Override configuration (the default file is `/etc/syslog.conf`).

`-D dir`
`--rcdir=dir`
: Override configuration directory (the default is `/etc/syslog.d`).

`-P file`
`--pidfile=file`
: Override pidfile (the default file is `/var/run/syslogd.pid`).

`-n`
`--no-detach`
: Do not enter daemon mode.

`-d`
`--debug` Print debug information (implies `-n`).

`-p file`
`--socket=file`
: Override default UNIX domain socket `/dev/log`.

`-a socket` Add UNIX socket to listen. An unlimited number of sockets is allowed.

`-r`
`--inet` Receive remote messages via Internet domain socket. Without this option no remote massages are received, since there is no listening socket. Yet sockets for forwarding are created on the fly as needed, which might cause performance issues on busy systems.

`-b address`
`--bind=address`
: Restrict the listening Internet domain socket to a single address. The default (given the use of `-r`) is a wildcard address, implying that the server listens at every available address. Any name will be resolved, and the lookup result will depend on the options `-4`, `-6`, and `--ipany`.

`--no-unixaf`
: Do not listen on UNIX domain sockets (overrides `-a` and `-p`).

`--no-klog`
: Do not listen to the kernel log device `/dev/klog`.

`--ipany` Allow both address families: IPv4 and IPv6.

`-4`
`--ipv4` Use only IPv4 for Internet domain sockets.

`-6`
`--ipv6` Use only IPv6 for Internet domain sockets.

`--no-forward`
: Do not forward any messages (overrides `-h`). This disables even temporary creation of forwarding sockets, an ability which is otherwise active when the option `-r` is left out.

`-h`
`--hop` Forward messages from remote hosts.

`-m` *`interval`*
`--mark=`*`interval`*
: Specify timestamp interval expressed in minutes (0 for no timestamping).

`-l` *`hostlist`*
: Log hosts in *hostlist* by their hostname. Multiple lists are allowed.

`-s` *`domainlist`*
: List of domains which should be stripped from the FQDN of hosts before logging their name. Multiple lists are allowed.

`-T`
`--local-time`
: Ignore any time contained in a received message. In its stead, record the time of reception on the local system. This circumvents problems caused by remote hosts with skewed clocks.

19.1 Configuration file

`syslogd` reads its configuration file when it starts up and whenever it receives a hangup signal. The `syslog.conf` file is the main configuration file for the `syslogd` program. In addition, the server looks below the directory `syslog.d/` for further configuration files, making it easy to arrange a common set of logging conventions in `syslog.conf`, augmented by system and service specific drop-in configuration in `syslog.d/`.

Each configuration file consists of lines with two fields: a *selector* field which specifies the types of messages and priorities to which the line applies, and an *action* field which specifies the action to be taken if a message `syslogd` receives matches the selection criteria. The selector field is separated from the action field by one or more tab or space characters. A rule can be split in several lines if all lines except the last are terminated with a backslash '\'.

There are two exceptional forms of line content. The first is the *tagged selector*, and the second is a comment. The latter begins with an octothorp ('`#`'), also called hash, and continues until end-of-line.

A tagged selector commences with an exclamation mark, as in '!name', or with a shebang, like '#! name', and continues with a program name, a *tag* in the sense used by logger. It has the effect of applying the following configuration rules only to messages submitted with the specified tag. This selection remains in effect until another tag is selected, or until it is reset by means of stating the program name as an asterisque '*'.

The selector fields are encoded as a facility, followed by a period ('.'), and a level, with no intervening white-space. The facility as well as the level are case insensitive.

The facility describes the part of the system generating the message, and is one of the following keywords: 'auth', 'authpriv', 'cron', 'daemon', 'kern', 'lpr', 'mail', 'mark', 'news', 'syslog', 'user', 'uucp' and 'local0' through 'local7'. These keywords (with the exception of 'mark') correspond to the similar 'LOG_' values specified to the 'openlog' and 'syslog' library routines. See Section "Syslog" in *The GNU C Library Reference Manual*, for details.

The level describes the severity of the message, and is a keyword from the following ordered list (higher to lower): 'emerg', 'alert', 'crit', 'err, warning', 'notice' and 'debug'. These keywords correspond to the similar 'LOG_' values specified to the syslog library routine.

See Section "syslog and vsyslog" in *The GNU C Library Reference Manual*, for a further descriptions of both the facility and level keywords and their significance.

If a received message matches the specified facility and is of the specified level (or a higher level), the action specified in the action field will be taken.

Multiple selectors may be specified for a single action by separating them with semicolon (';') characters. It is important to note, however, that each selector can modify the ones preceding it.

Multiple facilities may be specified for a single level by separating them with comma (',') characters.

An asterisk ('*') can be used to specify all facilities or all levels. Two asterisks ('**') specifie all facilities not named previously in the configuration file.

By default, a level applies to all messages with the same or higher level. The equal ('=') character can be prepended to a level to restrict this line of the configuration file to messages with the very same level.

An exclamation mark ('!') prepended to a level or the asterisk means that this line of the configuration file does not apply to the specified level (and higher ones). In conjunction with the equal sign, you can exclude single levels as well.

The special facility 'mark' receives a message at priority 'info' every 20 minutes. This is not enabled by a facility field containing an asterisk.

The special level 'none' disables a particular facility.

The action field of each line specifies the action to be taken when the selector field selects a message. There are five forms:

- A pathname (beginning with a leading slash). Selected messages are appended to the file.

  You may prepend a minus ('-') to the path to omit syncing the file after each message log. This can cause data loss at system crashes, but increases performance for programs which use logging extensively.

- A named pipe, beginning with a vertical bar ('|') followed by a pathname. The pipe must be created with mkfifo before syslogd reads its configuration file. This feature is especially useful for debugging.
- A hostname (preceded by an at ('@') sign). Selected messages are forwarded to syslogd on the named host.
- A comma separated list of users. Selected messages are written to those users if they are logged in.
- An asterisk. Selected messages are written to all logged-in users.

  Blank lines and lines whose first non-blank character is a hash ('#') character are ignored.

A configuration file might appear as follows:

```
# Log all kernel messages, authentication messages of
# level notice or higher and anything of level err or
# higher to the console.
# Don't log private authentication messages!
*.err;kern.*;auth.notice;authpriv.none  /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none          /var/log/messages

# The authpriv file has restricted access.
authpriv.*                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                  /var/log/maillog

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                                 *
*.emerg                                 @arpa.berkeley.edu

# Root and Eric get alert and higher messages.
*.alert                                 root,eric

# Simplify security auditing, by collecting sudo uses.
! sudo
*.info                                  /var/log/sudo

# Collect time server reports.
#! ntpd
*.*                                     /var/log/ntpd

# Stop selecting on message tags.
!*
```

```
# Save mail and news errors of level err and higher in a
# special file.
uucp,news.crit                                    /var/log/spoolerr
```

The effects of multiple selectors are sometimes not intuitive. For example '`mail.crit,*.err`' will select the '`mail`' facility messages at the level of '`err`' or higher, not at the level of '`crit`' or higher.

20 ftpd: FTP daemon

`ftpd` is the Internet File Transfer Protocol server process. The server uses the TCP protocol and listens at the port specified in the '`ftp`' service specification.

```
ftpd [option]...
```

`-4`
`--ipv4` Daemon uses only IPv4 addressing. Ignored in inetd mode.

`-6`
`--ipv6` Daemon uses only IPv6 addressing. Ignored in inetd mode.

`-A`
`--anonymous-only`
Only anonymous login is allowed.

`-a` *`auth`*
`--auth=`*`auth`*
Specify what authentication mechanism to use for incoming connections. Possible values are: '`kerberos`', '`kerberos5`', '`opie`', '`pam`', and '`default`'.

Anonymous logins will continue to work when this option is used, unless the user '`ftp`' is removed from the system.

`-D`
`--daemon` `ftpd` enters daemon-mode. That allows `ftpd` to be run without `inetd`.

`-d`
`--debug` Debugging information is written to the `syslog` using facility '`LOG_FTP`'.

`-l`
`--logging`
Each successful and failed ftp session is logged using `syslog` with a facility of '`LOG_FTP`'. If this option is specified twice, the retrieve (`get`), store (`put`), append, delete, make directory, remove directory and rename operations and their filename arguments are also logged.

`--non-rfc2577`
Do not follow the suggestion of RFC 2577 to suppress messages that could help an attacker to conduct user name enumeration. This option allows the server to return with an error message immediately upon receipt of a user name. Such information includes non-existence claims and expiration claims. The ideal mode would otherwise be to fake the relevance of asking for a password, and only thereafter report an invalid login.

`-p` *`pidfile`*
`--pidfile=`*`pidfile`*
Change default location of *pidfile*.

`-q`
`--no-version`
Quiet mode. No information about the version of the `ftpd` is given to the client.

`-T`
`--max-timeout`
A client may also request a different timeout period; the maximum period allowed may be set to timeout seconds with the `-T` option. The default limit is 2 hours.

`-t` *`timeout`*
`--timeout=`*`timeout`*
The inactivity timeout period is set to timeout seconds (the default is 15 minutes).

`-u` *`umask`*
`--umask=`*`umask`*
Set default umask, expressed in base 8.

The file `/etc/nologin` can be used to disable FTP access. If the file exists, `ftpd` displays it and exits. If the file `/etc/ftpwelcome` exists, `ftpd` prints it before issuing the '`ready`' message. If the file `/etc/motd` exists, `ftpd` prints it after a successful login.

If this server was compiled with PAM support, then any non-anonymous connection request will also be checked for settings pertaining to the PAM service '`ftp`', before finally being accepted.

Linux-PAM is particular in that it also provides a module '`pam_ftp.so`' influencing even anonymous access. By convention the present server relies on the functionality in that module when built on relevant systems. However, the module is known to be partially broken since ten years back, when one compares the claims in its manual page, so not all claimed trickery is available!

20.1 Standards

The FTP server currently supports the following FTP requests. The letter case of any request is ignored.

Request	**Description**
ABOR	abort previous command
ACCT	specify account (ignored)
ALLO	allocate storage (vacuously)
APPE	append to a file
CDUP	change to parent of current working directory
CWD	change working directory
DELE	delete a file
EPSV	extended passive transfer request
EPRT	specify data connection port
HELP	give help information
LIST	give list files in a directory ("ls -lgA")
LPRT	specify data connection port
LPSV	long passive transfer request
MKD	make a directory
MDTM	show last modification time of file

MODE	specify data transfer mode
NLST	give name list of files in directory
NOOP	do nothing
PASS	specify password
PASV	prepare for server-to-server transfer
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session
REST	restart incomplete transfer
RETR	retrieve a file
RMD	remove a directory
RNFR	specify rename-from file name
RNTO	specify rename-to file name
SITE	non-standard commands
SIZE	return size of file
STAT	return status of server
STOR	store a file
STOU	store a file with a unique name
STRU	specify data transfer structure
SYST	show operating system type of server system
TYPE	specify data transfer type
USER	specify user name
XCUP	change to parent of current working directory (deprecated)
XCWD	change working directory (deprecated)
XMKD	make a directory (deprecated)
XPWD	print the current working directory (deprecated)
XRMD	remove a directory (deprecated)

The following non-standard, or UNIX specific, commands are supported by the `SITE` request.

Request	Description
UMASK	change umask, e.g. `SITE UMASK 002`
IDLE	set idle-timer, e.g. `SITE IDLE 60`
CHMOD	change mode of a file, e.g. `SITE CHMOD0 0CHMOD1 1CHMOD2`
HELP	give help information.

The remaining FTP requests specified in RFC 959 are recognized, but not implemented. The extensions `MDTM`, `REST`, and `SIZE` are specified in RFC 3659, while `EPRT` and `EPSV` appear in RFC 2428, `LPRT` and `LPSV` in RFC 1639.

The ftp server will abort an active file transfer only when the `ABOR` command is preceded by a Telnet '`Interrupt Process`' (IP) signal and a Telnet '`Synch`' signal in the command Telnet stream, as described in Internet RFC 959. If a `STAT` command is received during a data transfer, preceded by a Telnet IP and Synch, transfer status will be returned.

`ftpd` interprets file names according to the globbing conventions used by `csh`. This allows users to utilize the metacharacters '`*?[]{}~`'.

The server applies the suggestions in RFC 2577, but the legacy behaviour with informational content in denials can be restored using the option `--non-rfc2577`.

20.2 Authentication

ftpd authenticates users according to four rules.

1. The login name must be in the password data base, /etc/passwd, and must not have a null password. In this case a password must be provided by the client before any file operations can be performed.
2. The login name must not appear in the file /etc/ftpusers.
3. The user must have a standard shell.
4. If the user name is ‘anonymous’ or ‘ftp’, an anonymous ftp account must be present in the password file (user ‘ftp’). In this case the user is allowed to log in by specifying any password (by convention an email address for the user should be used as the password).

A further access mechanism is provided by the file /etc/ftpchroot. A user mentioned therein will have all access confined to the subtree rooted at the home directory specified in /etc/passwd.

In the case of anonymous access, ftpd takes special measures to restrict the client's access privileges. The server always performs a chroot to the home directory of the ‘ftp’ user.

In order that system security is not breached, it is recommended that the ‘ftp’ subtree be constructed with care, following these rules:

~ftp Make the home directory owned by ‘root’ and not writable by anyone.

~ftp/bin Make this directory owned by ‘root’ and not writable by anyone (mode 555). The program ls must be present to support the list command, unless the server was compiled with libls support. This program should be mode 111.

~ftp/etc Make this directory owned by ‘root’ and not writable by anyone (mode 555). The files passwd and group must be present for the ls command to be able to produce owner names rather than numbers. The password field in passwd is not used, and should not contain real passwords. The file motd, if present, will be printed after a successful login. These files should be mode 444.

~ftp/pub Make this directory mode 777 and owned by ‘ftp’. Guests can then place files which are to be accessible via the anonymous account in this directory.

20.3 Configuration files

‘/etc/ftpchroot’
List of users to enclose in a chrooted directory. The anonymous user ‘ftp’ is always considered to be a member of this list, explicit or not.

‘/etc/ftpusers’
List of unwelcome/restricted users, always to be denied access.

‘/etc/ftpwelcome’
Welcome notice printed before server identification and any authentication exchange.

‘/etc/motd’
Welcome notice presented after completed login.

'/etc/nologin'

If present, the contents are displayed and all further access is refused.

20.4 File format of ftpusers and ftpchroot.

The files /etc/ftpusers and /etc/ftpchroot share a common file format. For better conformity with other implementations, each line is understood as consisting of fields separated by spaces, or by horizontal tabulators. Only the first non-empty field is examined at present. Both files are used for matching against a user name, desiring to use the FTP service.

Whenever the first printable character is a hash '#', the input line is taken as a comment, and is ignored. Lines lacking non-empty fields are likewise ignored.

A field consisting of a single at-sign '@', is treated as a wildcard and matches every input.

A field commencing with an at-sign '@' and then continuing with an identifier, is understood as giving the name of a group. Should this name exist in /etc/groups, and the user name be a member of this same group, then the user name matches.

In all other cases, the field is taken as the identifier of a user, with which the requesting user is compared for verbatim match.

It is worthwhile to observe from the above cases, that a single '@' on a line by itself in /etc/ftpchroot, will enforce chrooting upon every user allowed to access the FTP service. This gives a Draconian, protective configuration.

21 rexecd: server for rexec

rexecd is the server for the rexec routine. The server provides remote execution facilities with authentication based on user names and passwords. It passes error messages and notices to the syslog facility 'LOG_DAEMON'.

```
rexecd [option]...
```

rexecd listens for service requests at the port indicated in the 'exec' service specification. When a service request is received the following protocol is initiated:

1. The server reads characters from the socket up to a NUL ('\0') byte. The resultant string is interpreted as an ASCII number, base 10.
2. If the number received in step 1 is non-zero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine.
3. A NUL terminated user name of at most 16 characters is retrieved on the initial socket.
4. A NUL terminated, unencrypted password of at most 16 characters is retrieved on the initial socket.
5. A NUL terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
6. rexecd then validates the user as is done at login time and, if the authentication was successful, changes to the user's home directory, and establishes the user and group protections of the user. If any of these steps fail the connection is aborted with a diagnostic message returned.
7. A NUL byte is returned on the initial socket and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by rexecd.

21.1 Invoking

The only option is as follows:

-l
--logging
: Raise logging level for this service; use more than once for increased verbosity. The syslog facility in use is 'LOG_DAEMON'.

Should rexecd have been built with PAM support, it reads any setting specified for a service named 'rexec'.

21.2 Diagnostics

Except for the last one listed below, all diagnostic messages are returned on the initial socket, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

'username too long'
: The name is longer than 16 characters.

‘`password too long`’
: The password is longer than 16 characters.

‘`command too long`’
: The command line passed exceeds the size of the argument list (as configured into the system).

‘`Login incorrect.`’
: No password file entry for the user name existed.

‘`Password incorrect.`’
: The wrong password was supplied.

‘`No remote directory.`’
: The chdir command to the home directory failed.

‘`Try again.`’
: A fork by the server failed.

‘`<shellname>: ...`’
: The user’s login shell could not be started. This message is returned on the connection associated with the stderr, and is not ...

Note, that indicating ‘`Login incorrect`’ as opposed to ‘`Password incorrect`’ is a security breach which allows people to probe a system for users with null passwords.

22 rlogind: Remote login server

rlogind is the server for the rlogin client program (see Chapter 13 [rlogin invocation], page 32). The server provides a remote login facility with authentication based on privileged port numbers from trusted hosts, or using authentication according to a Kerberos protocol.

rlogind in daemon mode listens for service requests at the port indicated in the 'login' service specification. A common alternative is to have the super-server inetd listen at the same port, which then invokes rlogind as demand arises. In Kerberised mode, the port is either 'eklogin', or 'klogin', depending on preset encryption, or none.

The standard authentication procedure assumes the integrity of each client machine and of the connecting medium. This is insecure, since it transmits credentials in clear text, but is useful in an "open" environment. This weakness is reduced when running the service in Kerberised version, at the price of a larger complexity of the supporting infrastructure. Using an encrypting Kerberised service even avoids all clear text processing.

22.1 Invoking

The available options are as follows:

`-4`
`--ipv4` Accept only IPv4 connections in daemon mode.

`-6`
`--ipv6` Only IPv6 connections in daemon mode.

`-a`
`--verify-hostname`
: Ask hostname for verification.

`-d[`*max*`]`
`--daemon[=`*max*`]`
: Run in background daemon mode, optionally setting the maximal number of simultaneously running client sessions. The default limit is 10.

`-D[`*level*`]`
`--debug[=`*level*`]`
: Set debug level, not implemented.

`-l`
`--no-rhosts`
: Ignore client's .rhosts file.

`-L` *name*
`--local-domain=`*name*
: Set local domain name, to which the server host belongs. By default the domain is recovered from the canonical name of the host.

`-n`
`--no-keepalive`
: Do not set SO_KEEPALIVE on sockets. This decreases the ability to close lost connections to once active clients.

`-o`
`--allow-root`
Allow the root user to login, which is disallowed by default.

`-p` *port*
`--port=`*port*
Listen on given port. Applicable only in daemon mode.

`-r`
`--reverse-required`
Require reverse resolvability of remote host's numerical IP.

For sites requiring improved authentication, Kerberos authentication is a viable decision, and possibly even with encryption for enhanced integrity. Three additional options are available for an executable `rlogind` compiled with Kerberos support.

`-k`
`--kerberos`
Activate Kerberos authentication on all incoming requests.

`-S` *name*
`--server-principal=`*name*
Set Kerberos server name, overriding canonical hostname.

`-x`
`--encrypt`
Activate encryption of all data passed via the `rlogind` session. This may impact response time and CPU utilization, but provides increased security. Only for Kerberised mode of operation.

Should `rlogind` have been built with PAM support, it reads any setting specified for a service named either '`rlogin`' or '`krlogin`', the latter name for clients using Kerberised authentication.

22.2 Kerberos specific details

The option `-k` is mandatory for Kerberised operation mode, while addition of the option `-x` will also demand encryption of every request to this particular server.

`rlogind` will, in Kerberised operation mode, as default instantiate itself using the principal name '`host/canonical_name@DEFAULT_REALM`', a compound arranged from the running host's canonical name, and from the default realm configured for the system. Either of these can be overridden using the option `--server-principal`, as follows:

```
rlogind -k -S alias.server.our
rlogind --kerberos --server-principal=@NEW.REALM
rlogind -k -x -S rlogin/backup.ex.org@OUR.REALM
```

When overriding only the realm, with the option `-S`, an initial at-sign is mandatory.

22.3 Protocol details

When a service request is received, in non-Kerberised mode, the following protocol is initiated:

1. The server checks the client's source port. If the port is not in the range 512-1023, the server aborts the connection.
2. The server next checks the client's source address and requests the corresponding host name. If the hostname cannot be determined, the numerical representation of the host address is used. If the hostname is in the same domain as the server (according to the last two components of the domain name), or if the option `-a` is in effect, the address for the hostname is requested, verifying that the name and address correspond. Normal authentication is considered as failed, should this address verification fail.

Once the source port and address have been checked, `rlogind` proceeds with the authentication process as described in Chapter 23 [rshd invocation], page 63. The server then allocates a pseudo terminal, and manipulates file descriptors so that the slave half of the pseudo terminal becomes the stdin, stdout, and stderr for a login process. The login process is an instance of the `login` program, invoked with the option `-f` if authentication had succeeded. If automatic authentication had failed, the user is prompted to log in as if on a standard terminal line.

The parent of the login process manipulates the master side of the pseudo terminal, operating as an intermediary between the login process and the client instance of the rlogin program. In normal operation, the packet protocol described in '`PTY`' is invoked to provide flow control using *`C-S`*/*`C-Q`*, and to propagate interrupt signals to the remote program. The login process transmits the client terminal's baud rate, and its terminal type, as found in the environment variable `TERM`. The screen or window size of the terminal is requested from the client, and any later window size changes at the client's side are propagated to the pseudo terminal as well.

Transport-level keepalive messages are enabled unless the option `-n` was in effect when starting `rlogind`. The use of keepalive messages allows sessions to be timed out, should the client crash, or otherwise become unreachable.

See Section "ruserok" in *The GNU C Library Reference Manual*, for details.

22.4 Diagnostics

The exchange protocol states that a negotiation reaches a successful completion as soon as the server `rlogind` transmits back to the client a single null byte, marking the completion of all information exchange.

Error conditions are instead transmitted back to the client as a message containing an initial byte value 1, followed by a C-string indicating the cause of failure. All network connections are closed at the server side after this message. Some common messages follow:

'`Permission denied.`'

The client presented insufficient credentials, or the client's address is not sufficiently resolvable to pass the checks induced by options `-a` or `-r`.

'`Try again.`'

A fork by the server failed.

23 rshd: Remote shell server

The `rshd` server is the server for the `rcmd` routine and, consequently, for the `rsh` (see Chapter 14 [rsh invocation], page 35) program. The server provides remote execution facilities with authentication based on privileged port numbers from trusted hosts. The `rshd` server listens for service requests at the port indicated in the '`cmd`' service specification. When a service request is received the following protocol is initiated:

1. The server checks the client's source port. If the port is not in the range 512–1023, the server aborts the connection. However, this condition is not applied for Kerberized service.
2. The server reads characters from the socket up to a NUL ('`\0`') byte. The resultant string is interpreted as an ASCII number, base 10.
3. If the number received in step 2 is non-zero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine. The source port of this second connection is also in the range 512–1023.
4. The server checks the client's source address and requests the corresponding host name. If the hostname cannot be determined, the dot-notation representation of the host address is used. If the hostname is in the same domain as the server (according to the last two components of the domain name), or if the `-a` option is given, the addresses for the hostname are requested, verifying that the name and address correspond. If address verification fails, the connection is aborted with the message, '`Host address mismatch.`'
5. A null terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as the user identity on the client's machine.
6. A null terminated user name of at most 16 characters is retrieved on the initial socket. This user name is interpreted as a user identity to use on the server's machine.
7. A null terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
8. Rshd then validates the user using `ruserok`, which uses the file `/etc/hosts.equiv` and the `.rhosts` file found in the user's home directory. The `-l` option prevents `ruserok` from doing any validation based on the user's `.rhosts` file, unless the user is the superuser.
9. If the file `/etc/nologin` exists and the user is not the superuser, the connection is closed.
10. A null byte is returned on the initial socket and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rshd`.
11. Transport-level keepalive messages are enabled unless the `-n` option is present. The use of keepalive messages allows sessions to be timed out if the client crashes or becomes unreachable.
12. The `-L` option causes all successful accesses to be logged to `syslogd` (see Chapter 19 [syslogd invocation], page 48) as '`auth.info`' messages.

See Section "ruserok" in *The GNU C Library Reference Manual*, for details.

23.1 Invoking

The options are as follows:

`-a`
`--verify-hostname`
: Ask hostname for verification.

`-k`
`--kerberos`
: Use Kerberos authentication.

`-l`
`--no-rhosts`
: Ignore `.rhosts` file.

`-L`
`--log-sessions`
: Log successful logins.

`-n`
`--no-keepalive`
: Do not set SO_KEEPALIVE.

`-S` *name*
`--servername=`*name*
: Set Kerberos server name, overriding canonical hostname.

`-v`
`--vacuous`
: Fail any call asking for non-Kerberos authentication.

`-r`
`--reverse-required`
: Demand that the client's IP address be resolvable as a host name.

Should `rshd` have been built with PAM support, it reads any setting specified for a service named either '`rsh`' or '`krsh`', the latter name for clients seeking Kerberised authentication.

23.2 Diagnostics

Except for the last one listed below, all diagnostic messages are returned on the initial socket, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 10 above upon successful completion of all the steps prior to the execution of the login shell).

'`Locuser too long`'
: The name of the user on the client's machine is longer than 16 characters.

'`Ruser too long`'
: The name of the user on the remote machine is longer than 16 characters.

'`Command too long`'
: The command line passed exceeds the size of the argument list (as configured into the system).

'`Login incorrect`'
: No password file entry for the user name existed.

'`Remote directory`'
: The chdir command to the home directory failed.

'`Permission denied`'
: The authentication procedure described above failed, or address resolution was insufficient.

'`Can't make pipe.`'
: The pipe needed for the stderr, wasn't created.

'`Can't fork; try again.`'
: A fork by the server failed.

'`<shellname>: ...`'
: The user's login shell could not be started. This message is returned on the connection associated with the stderr, and is not preceded by a flag byte.

The authentication procedure used here assumes the integrity of each client machine and the connecting medium. This is insecure, but is useful in an "open" environment.

24 talkd: a server for communication between users

talkd is a server that notifies users that someone else wants to initiate a conversation. It acts as a repository of invitations, responding to requests by clients wishing to rendezvous for a conversation.

This implementation uses the newer protocol 'ntalk/udp', and is intended to be invoked by a super-server inetd at that datagram port. It is recommended that inetd launch talkd with ownership 'nobody:tty', or with 'tty:tty'. However, this works with ACL only if .talkrc can be assumed to be world readable for all users. This failing, the process ownership will need to be 'root:tty' if the ACL-mechanism is to be usable and trustworthy.

Keep in mind that this service is usable with IPv4 only, since the exchange protocol was conceived to handle only this particular address family. This fact is independent of the abilities of inetd.

Observe also that the server talkd depends on the name returned by hostname, for establishing connections between interested parties. A server talkd running on a multihomed host is not able to respond to invitations for a valid host name that differs from the name reported by hostname.

The present implementation offers ACL-mechanisms for fine grained access control.

24.1 Invoking

The following switches and options are available.

-a *file*
--acl=*file*
Read site-wide ACLs from *file*.

-d
--debug
Enable debugging.

-i *seconds*
--idle-timeout=*seconds*
Set idle timeout length

-l
--logging
Enable a somewhat enhanced logging verbosity, reporting attempted and dropped connections, as well as some more unexpected events that might arise.

-r *seconds*
--request-ttl=*seconds*
Set time-to-live length for requests.

-S
--strict-policy
Apply strict ACL policy on this system. This means that the site-wide ACL must provide explicit 'allow' rules for admitting traffic at all.

-t *seconds*
--timeout=*seconds*
Set timeout length.

24.2 Modus operandi

In normal operation, a client, the caller, initiates a rendezvous by sending a CTL_MSG of type 'LOOK_UP' to the server (see protocols/talkd.h). This causes the server to search its invitation tables to check whether an invitation currently exists for the caller (wanting to talk to the callee specified in the message). If the lookup fails, the caller then sends an 'ANNOUNCE' message causing the server to broadcast an announcement on the callee's login ports requesting contact. When the callee responds, the local server uses the recorded invitation to respond with the appropriate rendezvous address and the caller and callee client programs establish a stream connection through which the conversation takes place.

This implementation offers an additional mechanism, whereby a site-wide access control list can be used to limit service access in general. For any local user, i.e., present on the server's system, a further user owned file .talkrc is parsed, if at all present, in order to even further fine tune access to this particular user.

24.3 Access control in talkd

The server can be run in a mode with additional access control, beyond the legacy capabilities of ntalkd. This is activated using the option -a, or equivalently --acl.

The format of this access control list is shared with the user specific file .talkrc. Normally the site-wide setting operates with a default value 'allow', but specifying the option -S, or --strict-policy, changes this default action to 'deny'. In addition, the strict policy disables the possibility that an allowing action from the user specific ACL be able to override a denial resulting from the system-wide ACL setting.

As is usual, indentation, empty lines, and lines whose first printable character is the hash character, are all ignored. The general line format is

```
    action user-exp [net-exp ...]
```

Each active line must contain at least two fields: an action and a user-exp.

The first field, action, must be either of 'allow' and 'deny'. Any other value will lead to the line being ignored, but reported in the system log. Of course, the two values represent admitting and rejecting interpretations for the resulting rule.

The second field, user-exp, is a POSIX regular expression crafted to match user names. Remember that the regular expression would need anchors in order to test not only substrings.

It is important to note that in a site-wide ACL, the file selected by the switch -a, the expression user-exp is matched against the requested local user name, that of the callee.

While checking the callee's private ACL-file .talkrc, the matching of user-exp is done against the remote caller's name. Any other interpretation is plainly futile.

Each line may be augmented by a net list, containing one or more expressions net-exp. Each of these is either the simple word 'any', a numeric IPv4 address, or a full IPv4 address with an appended netmask. The effect is to restrict the applicability of the rule to the specified address range, or to set an explicit wildcard match 'any'. The absence of a net list is equivalent to specifying a single 'any'. The netmask can be specified as a CIDR mask length, or as an explicit address mask.

The actual evaluation is made separately for the site-wide ACL, and for the requested local user ACL, contained in the callee's private file .talkrc. This latter file must be a regular file and must be owned by the very same user, have his primary group ownership, and not be group or world writeable. Should any of these prerequisites be violated, the user's ACL is replaced by a single deny-all rule.

All rules in each set are evaluated, in the sense that whenever an expression `net-exp` matches the incoming IPv4 address, then the regular expression `user-exp` is tested for a match. That being the case, the corresponding action is recorded. The last match in each set determines the outcome in its category.

In the most common case, a system wide '`deny`' is overridden if the local user has specified at least one valid and applicable rule, admitting access. In the contrary case, where no admitting user rule could be established at all, then a resulting '`deny`', from a system wide ACL, will be used as the final action.

In strict policy mode, a site-wide '`deny`' is always final, ignoring any user's desire. The administrator must explicitly arrange some admitting rule, with an action '`allow`', and some suitable net list. Still, the individual user can arrange his private file for an even narrower selection of friends.

25 telnetd: Telnet server

```
telnetd [option]...
```

`-a authmode`
`--authmode=authmode`

Specify what mode to use for authentication. Allowed values are: '`none`', '`other`', '`user`', '`valid`', and '`off`'.

`-D[list]`
`--debug=[list]`

Set the debugging level. The argument is a comma separated list of these categories: '`options`', '`report`', '`netdata`', '`ptydata`', '`auth`', and '`encr`'. All these may be used in the form '`name[=level]`'. Omission of '`level`' implies the maximal possible debugging level for that particular category.

There is one additional category '`tcp`', which does not take an additional level indicator, but is instead equivalent to setting the socket option '`SO_DEBUG`' for debugging the complete traffic.

The output is written to the file `/tmp/telnet.debug`, and any new data is incrementally added as time passes.

`-E string`
`--exec-login=string`

Set program to be executed instead of `/bin/login`.

`-h`
`--no-hostinfo`

Do not print host information before login has been completed.

`-l[mode]`
`--linemode=[mode]`

Set line mode. An empty argument will force line read mode at all times. The only recognised value is otherwise '`nokludge`'.

`-n`
`--no-keepalive`

Disable TCP keep-alives.

`-S principal`
`--server-principal=principal`

Set principal name for the server, to be used in Kerberos authentication. The value *principal* can be set to provide full specification like '`srv.local@REALM`' and '`tnt/localhost@REALM`', where the first uses the standard prefix 'host/'. Or *principal* can override default settings in part only, like '`srv.local`', '`tnt/srv.local`', or '`@REALM`'.

`-U`
`--reverse-lookup`

Refuse connections from addresses that cannot be mapped back into a symbolic name. A client is accepted only if the IP address can be resolved as a host name, and the same name is resolvable to addresses among which the clients's address is included.

`-X` *authtype*
`--disable-auth-type=`*authtype*
: Disable the use of the given authentication type. Use this option multiple times if more than one type is to be disabled. Standard choices are '`null`', '`kerberos_v4`', and '`kerberos_v5`'.

25.1 Crafting an execution string.

The server `telnetd` contains a built-in execution string which invokes `login` with arguments suitable for the operating system at hand. This preset choice corresponds to the standard use case of the service. For specialized purposes this implementation also offers a command line option `-E`, or `--exec-login`, to override the built-in execution of `login`, thus allowing almost any choice of handler.

A custom execution string could look like

```
telnetd -h -E '/usr/local/sbin/avrop  %t %U'
```

The execution string must as its first part provide an absolute path to an executable file. After that may follow arbitrary additional arguments. For this latter part, `telnetd` offers some replacement tokens that dynamically are replaced by content. All are of the form `%<var>`, where '`<var>`' is a single letter from the following collection of selectors. A valid letter is called *variable*. The mark *conditional*, appearing below, indicates that the corresponding variable is conditionally assigned a value.

`%a` Returns '`ok`' whenever authentication is complete. *conditional*

`%d` Produces a time and date string.

`%h` Gives the remote host name in canonical form.

`%l` States the local host name, also in canonical form.

`%L` Returns the path of the pseudo terminal assigned to the client.

`%t` Gives the terminal device stripped of the leading '`/dev/`'.

`%T` States the terminal type, like '`xterm`'. *conditional*

`%u` Provides the authenticated user name. *conditional*

`%U` Returns the user name passed as an environment variable `USER` by the remote client software. The value is empty, should the environment not provide a value.

In addition, a conditional construct is able to take one action in case a variable has an assigned value, and optionally to take another action in the opposite case. The contruct is

```
%?<var>{true-stmt}[{false-stmt}]
```

The braces are here mandatory, while the brackets enclose the optional else-clause and are not included in actual use. The initial, motivating example, could thus be expanded to read

```
telnetd -h -E '/usr/local/sbin/avrop  %t %?a{%u krb5}{%U}'
```

In case authentication was completed as user '`sigge`', the execution string would resolve to

```
/usr/local/sbin/avrop  pts/1 sigge krb5
```

In all other cases the result would be

```
     /usr/local/sbin/avrop  pts/1 $USER
```

where `$USER` is the value of the corresponding environment variable and could possibly be empty.

26 tftpd: TFTP server

tftpd is intended to be invoked via inetd at all times.

Synopsis:

```
tftpd [options] [directory ...]
```

-g *group*
--group=*group*

Specify group membership of the process owner. This is used only along with the option -s, and replaces the group membership that comes from the process owner himself.

-l
--logging

Enable logging.

-n
--nonexistent

Supress negative acknowledgement of requests for nonexistent relative filenames.

-s *dir*
--secure-dir=*dir*

Let the serving process change its root directory to *dir* before attending to any requests. This directory is not observable by any client, but improves server isolation, since servable contents must be located below this chrooted directory *dir*.

-u *user*
--user=*user*

Specify the process owner for serving requests. Only relevant along with the option -s. The default name is 'nobody'.

26.1 Directory prefixes

In addition to options, an invocation of tftpd can specify an optional list of directory prefixes. These are approved of according to two principles:

- Relative pathnames are ignored.
- At most twenty prefixes are approved, the rest is discarded.

A request for a file is decided upon as a consequence of evaluating these criteria:

- Every file request containing the substring '/../' is denied, as is a file name beginning with '../'.
- Write requests must specify absolute locations.
- A file request, if specified as an *absolute* pathname, must begin with one of the approved directory prefixes, should at least one such prefix have been accepted.
- In the absence of a prefix collection, any absolute pathname is accepted, should the corresponding file exist.

- A file request, if specified as a *relative* name, will only be searched for below the acceptable prefixes, should at least one such prefix have been approved.
- A request for a relatively named file, is denied in the absence of approved directory prefixes.
- The resulting file must be world readable, or world writable, for a read request, or a write request, to succeed.

26.2 Use cases

The standard use case is an entry in `/etc/inetd.conf` like

```
tftp dgram udp4 wait root /usr/sbin/tftpd \
       tftpd /tftpboot /altboot
```

This would allow the TFTP client to use any of

```
get kernel
get /tftpboot/kernel
get kernel.alt
get /altboot/kernel.alt
get /etc/motd
```

given that `/tftpboot/kernel` and `/altboot/kernel.alt` exist. Observe that also `/etc/motd` is accessible, inspite there being no explicit mention of standard file locations.

A stronger mode of running a TFTP server is to use the 'secure mode', meaning that the serving process is running in a chrooted mode. Then a suitable configuration could be

```
tftp dgram udp4 wait root /usr/sbin/tftpd \
       tftpd --secure-dir=/srv/tftp-root  /tftpboot /altboot
```

Supposing the files `kernel` and `kernel.alt` to exist in the common directory `/srv/tftp-root/altboot/`, all the previously suggested client requests for a kernel would still be granted, but now any request for `/etc/motd` would be declined, and would get a reply 'File not found' back.

The chrooted setting is denying access outside of `/srv/tftp-root`, yet is not indicating this lock-in to the client, and is thus improving server isolation. Since neither of `-u` and `-g` were specified, the configuration reproduced above will in fact have the transmitting server process running with the default owner set to '`nobody:nogroup`'.

27 uucpd: Unix to Unix Copy relay daemon.

uucpd is a relay daemon responsible for accepting TCP transported connections for uucico. It is started by inetd, conducts any authentication, and then hands acceptable requests over to uucico.

```
uucpd [option]...
```

27.1 Options

There is a single, specific option available:

`-u location`
`--uucico=location`

Replace the hard coded location of uucico with the value specified as *location*.

27.2 Authentication steps.

Invocation is expected to be conducted by a protocol described exchange of user name and password; unfortunately in clear text. If those agree with existing local entries, then uucpd verifies that the stated user also has user shell location identical to the full file system location of uucico. Should that not be the case, the request is declined.

For this latter check, the option --uucico is useful when setting the configuration for inetd. It is recommended to wrap the invocation line of uucpd within a call to tcpd in the standard fashion.

Appendix A GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   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, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation 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. See `http://www.gnu.org/copyleft/`.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
  Copyright (C)  year  your name.
  Permission is granted to copy, distribute and/or modify this document
  under the terms of the GNU Free Documentation License, Version 1.3
  or any later version published by the Free Software Foundation;
  with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
  Texts.  A copy of the license is included in the section entitled ``GNU
  Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
    with the Invariant Sections being list their titles, with
    the Front-Cover Texts being list, and with the Back-Cover Texts
    being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Index