NAME

tcpd - access control facility for internet services

DESCRIPTION

The *tcpd* program can be set up to monitor incoming requests for *telnet, finger, ftp, exec, rsh, rlogin, tftp, talk, comsat* and other services that have a one-to-one mapping onto executable files.

The program supports both 4.3BSD-style sockets and System V.4-style TLI. Functionality may be limited when the protocol underneath TLI is not an internet protocol.

Operation is as follows: whenever a request for service arrives, the *inetd* daemon is tricked into running the *tcpd* program instead of the desired server. *tcpd* logs the request and does some additional checks. When all is well, *tcpd* runs the appropriate server program and goes away.

Optional features are: pattern-based access control, client username lookups with the RFC 931 etc. protocol, protection against hosts that pretend to have someone elses host name, and protection against hosts that pretend to have someone elses network address.

LOGGING

Connections that are monitored by *tcpd* are reported through the *syslog*(3) facility. Each record contains a time stamp, the client host name and the name of the requested service. The information can be useful to detect unwanted activities, especially when logfile information from several hosts is merged.

In order to find out where your logs are going, examine the syslog configuration file, usually /etc/syslog.conf.

ACCESS CONTROL

Optionally, *tcpd* supports a simple form of access control that is based on pattern matching. The access-control software provides hooks for the execution of shell commands when a pattern fires. For details, see the *hosts_access*(5) manual page.

HOST NAME VERIFICATION

The authentication scheme of some protocols (*rlogin, rsh*) relies on host names. Some implementations believe the host name that they get from any random name server; other implementations are more careful but use a flawed algorithm.

tcpd verifies the client host name that is returned by the address->name DNS server by looking at the host name and address that are returned by the name->address DNS server. If any discrepancy is detected, *tcpd* concludes that it is dealing with a host that pretends to have someone elses host name.

If the sources are compiled with -DPARANOID, *tcpd* will drop the connection in case of a host name/address mismatch. Otherwise, the hostname can be matched with the *PARANOID* wildcard, after which suitable action can be taken.

HOST ADDRESS SPOOFING

Optionally, *tcpd* disables source-routing socket options on every connection that it deals with. This will take care of most attacks from hosts that pretend to have an address that belongs to someone elses network. UDP services do not benefit from this protection. This feature must be turned on at compile time.

RFC 931

When RFC 931 etc. lookups are enabled (compile-time option) *tcpd* will attempt to establish the name of the client user. This will succeed only if the client host runs an RFC 931-compliant daemon. Client user name lookups will not work for datagram-oriented connections, and may cause noticeable delays in the case of connections from PCs.

EXAMPLES

The details of using *tcpd* depend on pathname information that was compiled into the program.

EXAMPLE 1

This example applies when *tcpd* expects that the original network daemons will be moved to an "other" place.

In order to monitor access to the *finger* service, move the original finger daemon to the "other" place and install tcpd in the place of the original finger daemon. No changes are required to configuration files.

```
# mkdir /other/place
# mv /usr/etc/in.fingerd /other/place
# cp tcpd /usr/etc/in.fingerd
```

The example assumes that the network daemons live in /usr/etc. On some systems, network daemons live in /usr/sbin or in /usr/libexec, or have no 'in.' prefix to their name.

EXAMPLE 2

This example applies when *tcpd* expects that the network daemons are left in their original place.

In order to monitor access to the *finger* service, perform the following edits on the *inetd* configuration file (usually */etc/inetd.conf* or */etc/inet/inetd.conf*):

finger stream tcp nowait nobody /usr/etc/in.fingerd in.fingerd

becomes:

finger stream tcp nowait nobody /some/where/tcpd in.fingerd

The example assumes that the network daemons live in /usr/etc. On some systems, network daemons live in /usr/sbin or in /usr/libexec, the daemons have no 'in.' prefix to their name, or there is no userid field in the inetd configuration file.

Similar changes will be needed for the other services that are to be covered by *tcpd*. Send a 'kill -HUP' to the *inetd*(8) process to make the changes effective. AIX users may also have to execute the 'inetimp' command.

EXAMPLE 3

In the case of daemons that do not live in a common directory ("secret" or otherwise), edit the *inetd* configuration file so that it specifies an absolute path name for the process name field. For example:

ntalk dgram udp wait root /some/where/tcpd /usr/local/lib/ntalkd

Only the last component (ntalkd) of the pathname will be used for access control and logging.

BUGS

Some UDP (and RPC) daemons linger around for a while after they have finished their work, in case another request comes in. In the inetd configuration file these services are registered with the *wait* option. Only the request that started such a daemon will be logged.

The program does not work with RPC services over TCP. These services are registered as *rpc/tcp* in the inetd configuration file. The only non-trivial service that is affected by this limitation is *rexd*, which is used by the *on(1)* command. This is no great loss. On most systems, *rexd* is less secure than a wildcard in /etc/hosts.equiv.

RPC broadcast requests (for example: *rwall, rup, rusers*) always appear to come from the responding host. What happens is that the client broadcasts the request to all *portmap* daemons on its network; each *portmap* daemon forwards the request to a local daemon. As far as the *rwall* etc. daemons know, the request comes from the local host.

FILES

The default locations of the host access control tables are:

/etc/hosts.allow
/etc/hosts.deny

SEE ALSO

hosts_access(5), format of the tcpd access control tables.
syslog.conf(5), format of the syslogd control file.
inetd.conf(5), format of the inetd control file.

AUTHORS

Wietse Venema (wietse@wzv.win.tue.nl),
Department of Mathematics and Computing Science,
Eindhoven University of Technology
Den Dolech 2, P.O. Box 513,
5600 MB Eindhoven, The Netherlands