NAME

hosts_options − host access control language extensions

DESCRIPTION

This document describes extensions to the language described in the hosts_access(5) document.

The extensible language uses the following format:

```
daemon_list : client_list : option : option ...
```

The first two fields are described in the hosts_access(5) manual page. The remainder of the rules is a list of zero or more options. Any ":" characters within options should be protected with a backslash.

An option is of the form "keyword" or "keyword value". Options are processed in the specified order. Some options are subjected to %<letter> substitutions. For the sake of backwards compatibility with earlier versions, an "=" is permitted between keyword and value.

LOGGING

severity mail.info

severity notice

Change the severity level at which the event will be logged. Facility names (such as mail) are optional, and are not supported on systems with older syslog implementations. The severity option can be used to emphasize or to ignore specific events.

ACCESS CONTROL

allow

deny Grant (deny) service. These options must appear at the end of a rule.

The *allow* and *deny* keywords make it possible to keep all access control rules within a single file, for example in the *hosts.allow* file.

To permit access from specific hosts only:

```
ALL: .friendly.domain: ALLOW
ALL: ALL: DENY
```

To permit access from all hosts except a few trouble makers:

```
ALL: .bad.domain: DENY
ALL: ALL: ALLOW
```

Notice the leading dot on the domain name patterns.

RUNNING OTHER COMMANDS

aclexec shell_command

Execute, in a child process, the specified shell command, after performing the %<letter> expansions described in the hosts_access(5) manual page. The command is executed with stdin, stdout and stderr connected to the null device, so that it won’t mess up the conversation with the client host. Example:

```
smtp : ALL : aclexec checkdnsbl %a
```

executes, in a background child process, the shell command "checkdnsbl %a" after replacing %a by the address of the remote host.

The connection will be allowed or refused depending on whether the command returns a true or false exit status.

spawn shell_command

Execute, in a child process, the specified shell command, after performing the %<letter> expansions described in the hosts_access(5) manual page. The command is executed with stdin, stdout and stderr connected to the null device, so that it won't mess up the conversation with the client host. Example:

spawn (/usr/sbin/safe_finger -l @%h | /usr/bin/mail root) &

executes, in a background child process, the shell command "safe_finger -l @%h | mail root" after replacing %h by the name or address of the remote host.

The example uses the "safe_finger" command instead of the regular "finger" command, to limit possible damage from data sent by the finger server. The "safe_finger" command is part of the daemon wrapper package; it is a wrapper around the regular finger command that filters the data sent by the remote host.

twist shell_command

Replace the current process by an instance of the specified shell command, after performing the %<letter> expansions described in the hosts_access(5) manual page. Stdin, stdout and stderr are connected to the client process. This option must appear at the end of a rule.

To send a customized bounce message to the client instead of running the real ftp daemon:

in.ftpd : ... : twist /bin/echo 421 Some bounce message

For an alternative way to talk to client processes, see the *banners* option below.

To run /some/other/in.telnetd without polluting its command-line array or its process environment:

in.telnetd : ... : twist PATH=/some/other; exec in.telnetd

Warning: in case of UDP services, do not twist to commands that use the standard I/O or the read(2)/write(2) routines to communicate with the client process; UDP requires other I/O primitives.

NETWORK OPTIONS

keepalive

Causes the server to periodically send a message to the client. The connection is considered broken when the client does not respond. The keepalive option can be useful when users turn off their machine while it is still connected to a server. The keepalive option is not useful for datagram (UDP) services.

linger number_of_seconds

Specifies how long the kernel will try to deliver not-yet delivered data after the server process closes a connection.

USERNAME LOOKUP

rfc931 [timeout_in_seconds]

Look up the client user name with the RFC 931 (TAP, IDENT, RFC 1413) protocol. This option is silently ignored in case of services based on transports other than TCP. It requires that the client system runs an RFC 931 (IDENT, etc.) -compliant daemon, and may cause noticeable delays with connections from non-UNIX clients. The timeout period is optional. If no timeout is specified a compile-time defined default value is taken.

MISCELLANEOUS

banners /some/directory
: Look for a file in ‘/some/directory´ with the same name as the daemon process (for example in.telnetd for the telnet service), and copy its contents to the client. Newline characters are replaced by carriage-return newline, and %<letter> sequences are expanded (see the hosts_access(5) manual page).

  The tcp wrappers source code distribution provides a sample makefile (Banners.Makefile) for convenient banner maintenance.

  Warning: banners are supported for connection-oriented (TCP) network services only.

nice [number]
: Change the nice value of the process (default 10). Specify a positive value to spend more CPU resources on other processes.

setenv name value
: Place a (name, value) pair into the process environment. The value is subjected to %<letter> expansions and may contain whitespace (but leading and trailing blanks are stripped off).

  Warning: many network daemons reset their environment before spawning a login or shell process.

umask 022
: Like the umask command that is built into the shell. An umask of 022 prevents the creation of files with group and world write permission. The umask argument should be an octal number.

user nobody

user nobody.kmem
: Assume the privileges of the "nobody" userid (or user "nobody", group "kmem"). The first form is useful with inetd implementations that run all services with root privilege. The second form is useful for services that need special group privileges only.

DIAGNOSTICS

When a syntax error is found in an access control rule, the error is reported to the syslog daemon; further options will be ignored, and service is denied.

SEE ALSO

hosts_access(5), the default access control language

AUTHOR

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