IPFilter Firewall User’s Guide

007–4599–001

CONTRIBUTORS

Written by Jean Wilson

Edited by Susan Wilkening

Production by Glen Traefald

Engineering contributions by Lindsey North

Record of Revision

Version	Description
001	November 2002 Original publication documenting the IPFilter firewall product that runs on IRIX systems.

Contents

Examples

About This Guide

IPFilter is a software package that provides certain Internet firewall protection on IRIX systems. This service uses a set of tools to control and configure the service.

IPFilter also provides Network Address Translation (NAT) support. This is done through a configuration file that is used to specify the mapping of addresses and ports.

This version of IPFilter is based on the current public doman version of IP Filter

Note: IPFilter should not be run with `ipfilterd` (part of the SGI `eoe.sw.ipgate` release). See the IPFilter release notes for details.

Related Publications

IPFilter is based on public domain software. As such, the documentation is also available online in several locations:

- The main IP Filter website: `http://coombs.anu.edu.au/ipfilter`
- IP Filter HOWTO document: `http://www.obfuscation.org/ipf/`
- IP Filter FAQ and other resources: `http://www.obfuscation.org/ipf/`

Obtaining Publications

You can obtain SGI documentation in the following ways:

- See the SGI Technical Publications Library at: `http://docs.sgi.com`. Various formats are available. This library contains the most recent and most comprehensive set of online books, release notes, man pages, and other information.
- If it is installed on your SGI system, you can use InfoSearch, an online tool that provides a more limited set of online books, release notes, and man pages. With an IRIX system, select **Help** from the Toolchest, and then select **InfoSearch**. Or you can type `infosearch` on a command line.

- You can also view release notes by typing either `grelnotes` or `relnotes` on a command line.
- You can also view man pages by typing `man` *title* on a command line.

Conventions

The following conventions are used throughout this document:

Convention	**Meaning**
`command`	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
variable	Italic typeface denotes variable entries and words or concepts being defined.
`user input`	This bold, fixed-space font denotes literal items that the user enters in interactive sessions. (Output is shown in nonbold, fixed-space font.)
[ ]	Brackets enclose optional portions of a command or directive line.
...	Ellipses indicate that a preceding element can be repeated.
`manpage`(*x*)	Man page section identifiers appear in parentheses after man page names.
GUI	This font denotes the names of graphical user interface (GUI) elements such as windows, screens, dialog boxes, menus, toolbars, icons, buttons, boxes, fields, and lists.

Reader Comments

If you have comments about the technical accuracy, content, or organization of this document, contact SGI. Be sure to include the title and document number of the manual with your comments. (Online, the document number is located in the front matter of the manual. In printed manuals, the document number is located at the bottom of each page.)

You can contact SGI in any of the following ways:

- Send e-mail to the following address:

  techpubs@sgi.com

- Use the Feedback option on the Technical Publications Library Web page:

  `http://docs.sgi.com`

- Contact your customer service representative and ask that an incident be filed in the SGI incident tracking system.

- Send mail to the following address:

  Technical Publications
  SGI
  1600 Amphitheatre Parkway, M/S 535
  Mountain View, California 94043–1351

- Send a fax to the attention of "Technical Publications" at +1 650 932 0801.

SGI values your comments and will respond to them promptly.

Chapter 1

About IPFilter

IPFilter is software that provides stateful packet filtering, enabling firewall and Network Address Translation (NAT) functionalities.

Rules are set up to specify which packets are denied or permited through the firewall. Keywords can be used to distinguish which interface a packet is associated with (either as a destination or as a result of route processing or a packet's receipt location).

IPFilter can be configured to filter using several IP header fields (described below). These filters are set when the rules are established:

- *Source and/or destination IP address.* Inverted hostnames and networks are also supported.
- *IP protocol.* Individual protocols can be specified, or more broad protocols, such as TCP/UDP. IPFilter matches either of the two protocols.
- *Fragments.* Fragmented packets can be selectively filtered out.
- *IP options.* It is possible to select packets based on which options are present and which options are **not** present.
- *Port number.* This is used with TCP/UDP IP protocols. Either the service name or the port number can be used.

IPFilter can also perform the following functions:

- Send back an ICMP error or TCP reset for denied packets
- Keep packet state information for TCP, UDP, and ICMP packet flows
- Keep fragment state information for any IP packet
- Act as a network address translator (NAT)
- Use redirection to set up transparent proxy connections
- Provide packet header details to user programs that use authentication information

A logging device is also available to track the functioning of IPFilter. This device supports logging of TCP/UDP/ICMP IP packet headers and the first 129 bytes of the packet when a packet is successfully passed through, when it is blocked and when a match is made for suspicious packets.

For a complete description of IPFilter functionality, see the IPFilter documentation and descriptions at `http://coombs.anu.edu.au/ipfilter`.

For a summary of IPFilter functionality and IRIX kernel information, see Chapter 2, "Setting Up IPFilter on IRIX Systems", page 3.

For details about the command line tools used with IPFilter, see Chapter 3, "IPFilter Commands and Tools", page 7.

Note: IPFilter should not be run with `ipfilterd` (part of the SGI `eoe.sw.ipgate` release). See the release notes provided with IPFilter for details.

Chapter 2

Setting Up IPFilter on IRIX Systems

Portions of the information in this chapter are derived from the *IPFilter Based Firewalls HOWTO*. See `http://www.obfuscation.org/ipf/` for a copy of that document and for complete details on establishing filtering rules.

This chapter provides an overview of the procedure for setting up, testing, and implementing filtering rules. The following sections are included in this chapter:

Rules Processing Overview

IPFilter uses configuration files that contain rules and rulesets. These rules determine what is filtered, how it is filtered, and other aspects of IPFilter use.

Rules are processed from top to bottom, one after the other. IPFilter does not stop comparing packets to rulesets after the first match is made. Unless interrupted, IPFilter goes through the entire ruleset and does not decide to pass or drop the packet until the last matching rule is checked. **The last matching rule always takes precedence.**

A simple rule file could be:

```
block in all
pass  in all
```

IPFilter checks the first rule, which is to block the packet. Then the second rule is checked, which is to pass in the packet. The last rule takes precedence, so the packet is passed on.

Several keywords can be used to establish filtering criteria. For example, the `quick` keyword can be added to any rule that makes the rule take action at that match. The following is an example of that keyword:

```
block in quick all
pass  in       all
```

When the `quick` keyword is used, the first rule is checked. Because that contains the `quick` keyword, action is taken (all packets are blocked). The second rule is never encountered and no packets are passed.

Several other keywords can be used to further refine the IPFilter rulesets. For example, the `on` keyword specifies that the data is coming in on a named interface. The `out` keyword can be used to filter outbound packets.

Different rules can be established to provide advanced security for your networks. For example, the `keep state` rule can track if a connection is established with TCP, UDP, or ICMP. When a packet arrives, an established state table is checked. The state table is a list of TCP, UDP or ICMP sessions that are automatically passed through the firewall, circumventing the entire ruleset. Because all TCP/IP sessions have a beginning, middle, and end, all that is needed is to filter the beginning of a TCP, UDP or ICMP session. *Keeping state* allows you to ignore the middle and end and focus on blocking or passing new sessions.

Rule groups allow you to write rulesets in a tree fashion, not as a linear list. Therefore, if a packet has no element that is involved in a rule, the rule can be skipped. Rules can be grouped by any common element that helps keep firewall flow moving: protocol, machine, netblocks or some other element specific to your environment.

See the *IPFilter Based Firewalls HOWTO* for complete details about keywords, rules, and rulesets.

IRIX Filtering Implementation

IPFilter uses configuration files to establish the rules used for filtering. IPFilter is controlled by using a start-up script that uses a `chkconfig` value and specific option values:

```
% /etc/init.d/ipf start | stop | reload
```

This script is executed when the host system is first booted. It creates the IPFilter devices, loads the `/etc/ipf.conf` and `/etc/ipnat.conf` rules files (which contain examples that are commented out), and it starts the logging daemon.

The script and other options can be used to start, stop, or reload IPFilter on your IRIX system.

The `on` or `off` value in the `chkconfig` file determines if IPFilter will be started and if the rules files will be loaded:

```
% chkconfig on|off ipfilter
```

The following files are used to set the options for different components of IPFilter on IRIX systems:

- `/etc/config/ipfilter.options`. The default value is `-Fa`, which flushes old rules.
- `/etc/config/ipnat.options`. The default value is `-CF`, which deletes active entries.
- `/etc/config/ipmon.options`. The default value is `-sn`, which creates a log, mapping addresses to names.

In addition to these files, the `/etc/ipf.conf` file contains filtering rules and the `/etc/ipnat.conf` file contains port and address mappings.

Filtering on IRIX Systems

The *IPFilter Based Firewalls HOWTO* describes in detail how to set up and use rules files. The following list describes some details used on IRIX systems.

1. Use the `mkfilters`(1) command to display basic entries for a configuration file:

   ```
   % mkfilters
   ```

2. Add rules to the basic `/etc/ipf.conf` file. See the *IPFilter Based Firewalls HOWTO* for details about creating rules.
3. Use the `ipftest`(1) command to test the rules file:

   ```
   % ipftest options
   ```

   Repeat steps 2 and 3 until the rules in `ipf.conf` are satisfactory.
4. If necessary, modify the `/etc/config/ipf.options` file and the `/etc/config/ipmon.options` file using a text editor such as `vi`.

When the testing is done, issue the following commands to start IPFilter:

```
% chkconfig ipfilter on
% /etc/init.d/ipf stop
```

```
% /etc/init.d/ipf start
```

Use `/etc/init.d/ipf reload` to reload the rules file after changing it.

Several commands can be used to help you troubleshoot the filtering process and the rules used for filtering. Use the `ipf -V` command to check the current state of IPFilter. Use the `ipfstat` command to check packet filter statistics and the `ipfstat -nio` command to view the filter list.

See Chapter 3, "IPFilter Commands and Tools", page 7 for a description of IPFilter commands and command usage.

Establishing Network Address Translation (NAT) on IRIX Systems

Network Address Translation (NAT) gives an administrator the ability to connect several computers through a common external interface. In addition, another common use of NAT is to take statically allocated blocks of addresses and map many computers into this smaller address space.

See the *IPFilter Based Firewalls HOWTO* for complete details about setting up maps for NAT.

NAT on IRIX Systems

Setting up NAT on IRIX systems is similar to setting up filtering.

1. Add mapping to the basic `/etc/ipnat.conf` file. See the *IPFilter Based Firewalls HOWTO* for details about creating maps.

2. Issue the following commands to start IPFilter with NAT:

   ```
   % chkconfig ipfilter on
   % /etc/init.d/ipf stop
   % /etc/init.d/ipf start
   ```

Chapter 3

IPFilter Commands and Tools

IPFilter has several commands that can be used to log information, configure IPFilter, and perform other administrative functions. The following groups of commands are available:

- `ipf`: reads in rules and appends them to the kernel's current list. See the `ipf`(4), `ipf`(5), and `ipf`(8) man pages for complete details.
- `ipfs`: saves and restores information for NAT and state tables. See the `ipfs`(8) man page for details.
- `ipfstat`: reports on packet filter statistics and filter lists. See the `ipfstat`(8) man page for details.
- `ipftest`: tests packet filter rules. See the `ipftest`(1) man page for details.
- `ipl`: gathers packet headers to log. See the `ipl`(4) man page for details.
- `ipmon`: checks for logged packets. See the `ipmon`(8) man page for details.
- `ipnat`: reads in rules and adds them to the kernel's current list of active NAT rules. See the `ipnat`(4), `ipnat`(5), and `ipnat`(8) man pages for details.
- `ipresend`: resends IP packets out to the network. See the `ipresend`(1) man page for details.
- `ipsend`: sends IP packets. See the `ipsend`(1) and `ipsend`(5) man pages for details.
- `mkfilters`: generates a minimal ruleset for IPFilter. See the `mkfilters`(1) man page for details.

The `iptest` command is documented in the *IPFilter Based Firewalls HOWTO* but it is not available on IRIX systems.

See the *IPFilter Based Firewalls HOWTO*, available from `http://www.obfuscation.org/ipf/`, for detailed examples of command usage.

Examples of Use

The following examples demonstrate the use of several IPFilter commands. Note that the output examples will be different from system to system.

Example 3-1 `ipf`(8) Example

The `ipf` command can be used to enable and disable filters, load rules files, and enable logging. It can also be used to display information about the currently activated system, as in this example:

```
% ipf -V
ipf: IPFilter: v3.4.27 (384)
Kernel: IPFilter: v3.4.27
Running: yes
Log Flags: 0 = none set
Default: pass all, Logging: available
Active list: 0
```

Example 3-2 Sample Rules File

Rules are stored in `/etc/ipf.conf`. The following is an example of a rules file.

```
block in           on tun0
block in     quick on tun0 from 192.168.0.0/16 to any
block in     quick on tun0 from 172.16.0.0/12 to any
block in     quick on tun0 from 10.0.0.0/8 to any
block in     quick on tun0 from 127.0.0.0/8 to any
block in     quick on tun0 from 0.0.0.0/8 to any
block in     quick on tun0 from 169.254.0.0/16 to any
block in     quick on tun0 from 192.0.2.0/24 to any
block in     quick on tun0 from 204.152.64.0/23 to any
block in     quick on tun0 from 224.0.0.0/3 to any
block in log quick on tun0 from 20.20.20.0/24 to any
block in log quick on tun0 from any to 20.20.20.0/32
block in log quick on tun0 from any to 20.20.20.255/32
pass  out quick on tun0 proto tcp/udp from 20.20.20.1/32 to any keep state
pass  out quick on tun0 proto icmp    from 20.20.20.1/32 to any keep state
```

Example 3-3 `ipftest`(1) Example

The `ipftest` command allows you to check a rules file offline by entering packet information from the keyboard or from an input file:

```
% ipftest -r /etc/ipf.conf
```

The following output appears:

```
opening rule file "/etc/ipf.conf"
in on ec0 xxx.xxx.xx.198, xxx.xxx.xx.224
input: in on ec0 xxx.xxx.xx.198, xxx.xxx.xx.224
pass ip 20(20) 0 255.255.255.255 > xxx.xxx.xx.224
```

Example 3-4 `ipfstat`(8) Example

The `ipfstat` command displays kernel statistics. Note that no output appears for any `ipfstat` command unless this command is issued by a privileged user.

```
% ipfstat
```

The following output appears:

```
input packets:    blocked 30 passed 1887 nomatch 159 counted 0 short 0
output packets:   blocked 0 passed 449 nomatch 262 counted 0 short 0
input packets logged:  blocked 30 passed 23
output packets logged: blocked 0 passed 22
packets logged:        input 0 output 0
log failures:          input 0 output 0
fragment state(in):    kept 0  lost 0
fragment state(out):   kept 0  lost 0
packet state(in):      kept 2  lost 0
packet state(out):     kept 0  lost 0
ICMP replies:   0      TCP RSTs sent:  1
Invalid source(in):    0
Result cache hits(in): 172     (out):  159
IN Pullups succeeded:  0       failed: 0
OUT Pullups succeeded: 0       failed: 0
Fastroute successes:   1       failures:       0
TCP cksum fails(in):   0       (out):  0
Packet log flags set: (0)
        none
```

This command can also show the current rules:

```
% ipfstat -nio
```

Example 3-5 `ipmon`(8) Example

The `ipmon` command logs packets to `syslog` or to another specified file. The logged packets are those which match rules that contain the `log` keyword or which match the decision type given when the `ipf -l` *decision-type* command is issued.

The `/etc/init.d/ipf` startup script starts `ipmon` in the background with the `-sn` options on the command to log to `syslog`. The logging includes only header information unless it has been invoked with the `-b` option and the rule indicates `log body`.

```
% tail /var/adm/SYSLOG
2136]: 16:00:00.062786 ef0 @0:2 b starfish.sgi.com[XX.XX.XX.224] ->
seaweed.sgi.com[XX.XX.XX.198] PR icmp len 20 84 icmp echo/0 IN
```

Index