NAME

iptables/ip6tables — administration tool for IPv4/IPv6 packet filtering and NAT

SYNOPSIS

iptables [**−t** *table*] {**−A**|**−C**|**−D**} *chain rule-specification*

ip6tables [**−t** *table*] {**−A**|**−C**|**−D**} *chain rule-specification*

iptables [**−t** *table*] **−I** *chain* [*rulenum*] *rule-specification*

iptables [**−t** *table*] **−R** *chain rulenum rule-specification*

iptables [**−t** *table*] **−D** *chain rulenum*

iptables [**−t** *table*] **−S** [*chain* [*rulenum*]]

iptables [**−t** *table*] {**−F**|**−L**|**−Z**} [*chain* [*rulenum*]] [*options...*]

iptables [**−t** *table*] **−N** *chain*

iptables [**−t** *table*] **−X** [*chain*]

iptables [**−t** *table*] **−P** *chain target*

iptables [**−t** *table*] **−E** *old-chain-name new-chain-name*

rule-specification = [*matches...*] [*target*]

match = **−m** *matchname* [*per-match-options*]

target = **−j** *targetname* [*per−target−options*]

DESCRIPTION

Iptables and **ip6tables** are used to set up, maintain, and inspect the tables of IPv4 and IPv6 packet filter rules in the Linux kernel. Several different tables may be defined. Each table contains a number of built-in chains and may also contain user-defined chains.

Each chain is a list of rules which can match a set of packets. Each rule specifies what to do with a packet that matches. This is called a ‘target’, which may be a jump to a user-defined chain in the same table.

TARGETS

A firewall rule specifies criteria for a packet and a target. If the packet does not match, the next rule in the chain is examined; if it does match, then the next rule is specified by the value of the target, which can be the name of a user-defined chain, one of the targets described in **iptables−extensions**(8), or one of the special values **ACCEPT**, **DROP** or **RETURN**.

ACCEPT means to let the packet through. **DR OP** means to drop the packet on the floor. **RETURN** means stop traversing this chain and resume at the next rule in the previous (calling) chain. If the end of a built-in chain is reached or a rule in a built-in chain with target **RETURN** is matched, the target specified by the chain policy determines the fate of the packet.

TABLES

There are currently five independent tables (which tables are present at any time depends on the kernel configuration options and which modules are present).

−t, **−−table** *table*

This option specifies the packet matching table which the command should operate on. If the kernel is configured with automatic module loading, an attempt will be made to load the appropriate module for that table if it is not already there.

The tables are as follows:

filter: This is the default table (if no −t option is passed). It contains the built-in chains **INPUT** (for packets destined to local sockets), **FORWARD** (for packets being routed through the box), and **OUTPUT** (for locally-generated packets).

nat: This table is consulted when a packet that creates a new connection is encountered. It consists of four built-ins: **PREROUTING** (for altering packets as soon as they come in),

INPUT (for altering packets destined for local sockets), **OUTPUT** (for altering locally-generated packets before routing), and **POSTROUTING** (for altering packets as they are about to go out). IPv6 NAT support is available since kernel 3.7.

mangle:
This table is used for specialized packet alteration. Until kernel 2.4.17 it had two built-in chains: **PREROUTING** (for altering incoming packets before routing) and **OUTPUT** (for altering locally-generated packets before routing). Since kernel 2.4.18, three other built-in chains are also supported: **INPUT** (for packets coming into the box itself), **FORWARD** (for altering packets being routed through the box), and **POSTROUTING** (for altering packets as they are about to go out).

raw: This table is used mainly for configuring exemptions from connection tracking in combination with the NOTRACK target. It registers at the netfilter hooks with higher priority and is thus called before ip_conntrack, or any other IP tables. It provides the following built-in chains: **PREROUTING** (for packets arriving via any network interface) **OUTPUT** (for packets generated by local processes)

security:
This table is used for Mandatory Access Control (MAC) networking rules, such as those enabled by the **SECMARK** and **CONNSECMARK** targets. Mandatory Access Control is implemented by Linux Security Modules such as SELinux. The security table is called after the filter table, allowing any Discretionary Access Control (DAC) rules in the filter table to take effect before MAC rules. This table provides the following built-in chains: **INPUT** (for packets coming into the box itself), **OUTPUT** (for altering locally-generated packets before routing), and **FORWARD** (for altering packets being routed through the box).

OPTIONS

The options that are recognized by **iptables** and **ip6tables** can be divided into several different groups.

COMMANDS

These options specify the desired action to perform. Only one of them can be specified on the command line unless otherwise stated below. For long versions of the command and option names, you need to use only enough letters to ensure that **iptables** can differentiate it from all other options.

−A, **−−append** *chain rule-specification*
Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.

−C, **−−check** *chain rule-specification*
Check whether a rule matching the specification does exist in the selected chain. This command uses the same logic as **−D** to find a matching entry, but does not alter the existing iptables configuration and uses its exit code to indicate success or failure.

−D, **−−delete** *chain rule-specification*
−D, **−−delete** *chain rulenum*
Delete one or more rules from the selected chain. There are two versions of this command: the rule can be specified as a number in the chain (starting at 1 for the first rule) or a rule to match.

−I, **−−insert** *chain* [*rulenum*] *rule-specification*
Insert one or more rules in the selected chain as the given rule number. So, if the rule number is 1, the rule or rules are inserted at the head of the chain. This is also the default if no rule number is specified.

−R, **−−replace** *chain rulenum rule-specification*
Replace a rule in the selected chain. If the source and/or destination names resolve to multiple addresses, the command will fail. Rules are numbered starting at 1.

−L, **−−list** [*chain*]
List all rules in the selected chain. If no chain is selected, all chains are listed. Like every other iptables command, it applies to the specified table (filter is the default), so NAT rules get listed by
iptables −t nat −n −L
Please note that it is often used with the **−n** option, in order to avoid long reverse DNS lookups. It is legal to specify the **−Z** (zero) option as well, in which case the chain(s) will be atomically listed and zeroed. The exact output is affected by the other arguments given. The exact rules are suppressed until you use
iptables −L −v
or **iptables−save**(8).

−S, **−−list−rules** [*chain*]
Print all rules in the selected chain. If no chain is selected, all chains are printed like iptables-save. Like every other iptables command, it applies to the specified table (filter is the default).

−F, **−−flush** [*chain*]
Flush the selected chain (all the chains in the table if none is given). This is equivalent to deleting all the rules one by one.

−Z, **−−zero** [*chain* [*rulenum*]]
Zero the packet and byte counters in all chains, or only the given chain, or only the given rule in a chain. It is legal to specify the **−L**, **−−list** (list) option as well, to see the counters immediately before they are cleared. (See above.)

−N, **−−new−chain** *chain*
Create a new user-defined chain by the given name. There must be no target of that name already.

−X, **−−delete−chain** [*chain*]
Delete the optional user-defined chain specified. There must be no references to the chain. If there are, you must delete or replace the referring rules before the chain can be deleted. The chain must be empty, i.e. not contain any rules. If no argument is given, it will attempt to delete every non-builtin chain in the table.

−P, **−−policy** *chain target*
Set the policy for the built-in (non-user-defined) chain to the given target. The policy target must be either **ACCEPT** or **DROP**.

−E, **−−rename−chain** *old−chain new−chain*
Rename the user specified chain to the user supplied name. This is cosmetic, and has no effect on the structure of the table.

−h Help. Give a (currently very brief) description of the command syntax.

PARAMETERS

The following parameters make up a rule specification (as used in the add, delete, insert, replace and append commands).

−4, **−−ipv4**
This option has no effect in iptables and iptables-restore. If a rule using the **−4** option is inserted with (and only with) ip6tables-restore, it will be silently ignored. Any other uses will throw an error. This option allows IPv4 and IPv6 rules in a single rule file for use with both iptables-restore and ip6tables-restore.

−6, **−−ipv6**
If a rule using the **−6** option is inserted with (and only with) iptables-restore, it will be silently ignored. Any other uses will throw an error. This option allows IPv4 and IPv6 rules in a single rule file for use with both iptables-restore and ip6tables-restore. This option has no effect in ip6tables and ip6tables-restore.

[**!**] **−p**, **−−protocol** *protocol*
The protocol of the rule or of the packet to check. The specified protocol can be one of **tcp**, **udp**, **udplite**, **icmp**, **icmpv6**,**esp**, **ah**, **sctp**, **mh** or the special keyword "**all**", or it can be a numeric

value, representing one of these protocols or a different one. A protocol name from /etc/protocols is also allowed. A "!" argument before the protocol inverts the test. The number zero is equivalent to **all**. "**all**" will match with all protocols and is taken as default when this option is omitted. Note that, in ip6tables, IPv6 extension headers except **esp** are not allowed. **esp** and **ipv6−nonext** can be used with Kernel version 2.6.11 or later. The number zero is equivalent to **all**, which means that you cannot test the protocol field for the value 0 directly. To match on a HBH header, even if it were the last, you cannot use **−p 0**, but always need **−m hbh**.

[**!**] **−s**, **−−source** *address*[*/mask*][**,**...]
Source specification. *Address* can be either a network name, a hostname, a network IP address (with */mask*), or a plain IP address. Hostnames will be resolved once only, before the rule is submitted to the kernel. Please note that specifying any name to be resolved with a remote query such as DNS is a really bad idea. The *mask* can be either an ipv4 network mask (for iptables) or a plain number, specifying the number of 1's at the left side of the network mask. Thus, an iptables mask of *24* is equivalent to *255.255.255.0*. A "!" argument before the address specification inverts the sense of the address. The flag **−−src** is an alias for this option. Multiple addresses can be specified, but this will **expand to multiple rules** (when adding with −A), or will cause multiple rules to be deleted (with −D).

[**!**] **−d**, **−−destination** *address*[*/mask*][**,**...]
Destination specification. See the description of the **−s** (source) flag for a detailed description of the syntax. The flag **−−dst** is an alias for this option.

−m, **−−match** *match*
Specifies a match to use, that is, an extension module that tests for a specific property. The set of matches make up the condition under which a target is invoked. Matches are evaluated first to last as specified on the command line and work in short-circuit fashion, i.e. if one extension yields false, evaluation will stop.

−j, **−−jump** *target*
This specifies the target of the rule; i.e., what to do if the packet matches it. The target can be a user-defined chain (other than the one this rule is in), one of the special builtin targets which decide the fate of the packet immediately, or an extension (see **EXTENSIONS** below). If this option is omitted in a rule (and **−g** is not used), then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented.

−g, **−−goto** *chain*
This specifies that the processing should continue in a user specified chain. Unlike the −−jump option return will not continue processing in this chain but instead in the chain that called us via −−jump.

[**!**] **−i**, **−−in−interface** *name*
Name of an interface via which a packet was received (only for packets entering the **INPUT**, **FORWARD** and **PREROUTING** chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

[**!**] **−o**, **−−out−interface** *name*
Name of an interface via which a packet is going to be sent (for packets entering the **FORWARD**, **OUTPUT** and **POSTROUTING** chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

[**!**] **−f**, **−−fragment**
This means that the rule only refers to second and further IPv4 fragments of fragmented packets. Since there is no way to tell the source or destination ports of such a packet (or ICMP type), such a packet will not match any rules which specify them. When the "!" argument precedes the "−f" flag, the rule will only match head fragments, or unfragmented packets. This option is IPv4 specific, it is not available in ip6tables.

−c, **−−set−counters** *packets bytes*
This enables the administrator to initialize the packet and byte counters of a rule (during **INSERT**, **APPEND**, **REPLACE** operations).

OTHER OPTIONS
The following additional options can be specified:

−v, **−−verbose**
Verbose output. This option makes the list command show the interface name, the rule options (if any), and the TOS masks. The packet and byte counters are also listed, with the suffix 'K', 'M' or 'G' for 1000, 1,000,000 and 1,000,000,000 multipliers respectively (but see the **−x** flag to change this). For appending, insertion, deletion and replacement, this causes detailed information on the rule or rules to be printed. **−v** may be specified multiple times to possibly emit more detailed debug statements.

−w, **−−wait** [*seconds*]
Wait for the xtables lock. To prevent multiple instances of the program from running concurrently, an attempt will be made to obtain an exclusive lock at launch. By default, the program will exit if the lock cannot be obtained. This option will make the program wait (indefinitely or for optional *seconds*) until the exclusive lock can be obtained.

−W, **−−wait-interval** *microseconds*
Interval to wait per each iteration. When running latency sensitive applications, waiting for the xtables lock for extended durations may not be acceptable. This option will make each iteration take the amount of time specified. The default interval is 1 second. This option only works with **−w**.

−n, **−−numeric**
Numeric output. IP addresses and port numbers will be printed in numeric format. By default, the program will try to display them as host names, network names, or services (whenever applicable).

−x, **−−exact**
Expand numbers. Display the exact value of the packet and byte counters, instead of only the rounded number in K's (multiples of 1000) M's (multiples of 1000K) or G's (multiples of 1000M). This option is only relevant for the **−L** command.

−−line−numbers
When listing rules, add line numbers to the beginning of each rule, corresponding to that rule's position in the chain.

−−modprobe=*command*
When adding or inserting rules into a chain, use *command* to load any necessary modules (targets, match extensions, etc).

LOCK FILE

iptables uses the */run/xtables.lock* file to take an exclusive lock at launch.

The **XTABLES_LOCKFILE** environment variable can be used to override the default setting.

MATCH AND TARGET EXTENSIONS

iptables can use extended packet matching and target modules. A list of these is available in the **iptables−extensions**(8) manpage.

DIAGNOSTICS

Various error messages are printed to standard error. The exit code is 0 for correct functioning. Errors which appear to be caused by invalid or abused command line parameters cause an exit code of 2, and other errors cause an exit code of 1.

BUGS

Bugs? What's this? ;-) Well, you might want to have a look at http://bugzilla.netfilter.org/

COMPATIBILITY WITH IPCHAINS

This **iptables** is very similar to ipchains by Rusty Russell. The main difference is that the chains **INPUT** and **OUTPUT** are only traversed for packets coming into the local host and originating from the local host respectively. Hence every packet only passes through one of the three chains (except loopback traffic, which involves both INPUT and OUTPUT chains); previously a forwarded packet would pass through all three.

The other main difference is that **−i** refers to the input interface; **−o** refers to the output interface, and both are available for packets entering the **FORWARD** chain.

The various forms of NAT have been separated out; **iptables** is a pure packet filter when using the default 'filter' table, with optional extension modules. This should simplify much of the previous confusion over the combination of IP masquerading and packet filtering seen previously. So the following options are handled differently:
−j MASQ
−M −S
−M −L
There are several other changes in iptables.

SEE ALSO

iptables−apply(8), **iptables−save**(8), **iptables−restore**(8), **iptables−extensions**(8),

The packet-filtering-HOWTO details iptables usage for packet filtering, the NAT-HOWTO details NAT, the netfilter-extensions-HOWTO details the extensions that are not in the standard distribution, and the netfilter-hacking-HOWTO details the netfilter internals.
See **http://www.netfilter.org/**.

AUTHORS

Rusty Russell originally wrote iptables, in early consultation with Michael Neuling.

Marc Boucher made Rusty abandon ipnatctl by lobbying for a generic packet selection framework in iptables, then wrote the mangle table, the owner match, the mark stuff, and ran around doing cool stuff everywhere.

James Morris wrote the TOS target, and tos match.

Jozsef Kadlecsik wrote the REJECT target.

Harald Welte wrote the ULOG and NFQUEUE target, the new libiptc, as well as the TTL, DSCP, ECN matches and targets.

The Netfilter Core Team is: Jozsef Kadlecsik, Pablo Neira Ayuso, Eric Leblond, Florian Westphal and Arturo Borrero Gonzalez. Emeritus Core Team members are: Marc Boucher, Martin Josefsson, Yasuyuki Kozakai, James Morris, Harald Welte and Rusty Russell.

Man page originally written by Herve Eychenne <rv@wallfire.org>.

VERSION

This manual page applies to iptables/ip6tables 1.8.7.