Contents

How To List and Delete Iptables Firewall Rules

Posted August 14, 2015 1.9m FIREWALL NETWORKING SECURITY

By Mitchell Anicas
Become an author

Introduction

Iptables is a firewall that plays an essential role in network security for most Linux systems. While many iptables tutorials will teach you how to create firewall rules to secure your server, this one will focus on a different aspect of firewall management: listing and deleting rules.

In this tutorial, we will cover how to do the following iptables tasks:

- List rules
- Clear Packet and Byte Counters

- Delete rules
- Flush chains (delete all rules in a chain)
- Flush all chains and tables, delete all chains, and accept all traffic

Note: When working with firewalls, take care not to lock yourself out of your own server by blocking SSH traffic (port 22, by default). If you lose access due to your firewall settings, you may need to connect to it via the console to fix your access. Once you are connected via the console, you can change your firewall rules to allow SSH access (or allow all traffic). If your saved firewall rules allow SSH access, another method is to reboot your server.

Prerequisites

Before you start using this tutorial, you should have a separate, non-root superuser account—a user with sudo privileges—set up on your server. If you need to set this up, follow the appropriate guide:

- Initial Server Setup with Ubuntu 14.04
- Initial Server Setup with CentOS 6

Let's look at how to list rules first. There are two different ways to view your active iptables rules: in a table or as a list of rule specifications. Both methods provide roughly the same information in different formats.

List Rules by Specification

To list out all of the active iptables rules by specification, run the `iptables` command with the `-S` option:

```
$ sudo iptables -S
```

```
Example: Rule Specification Listing
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT ACCEPT
-N ICMP
-N TCP
-N UDP
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m conntrack --ctstate INVALID -j DROP
-A INPUT -p udp -m conntrack --ctstate NEW -j UDP
-A INPUT -p tcp -m tcp --tcp-flags FIN,SYN,RST,ACK SYN -m conntrack --ctstate NEW -j TCP
-A INPUT -p icmp -m conntrack --ctstate NEW -j ICMP
-A INPUT -p udp -j REJECT --reject-with icmp-port-unreachable
-A INPUT -p tcp -j REJECT --reject-with tcp-reset
-A INPUT -j REJECT --reject-with icmp-proto-unreachable
-A TCP -p tcp -m tcp --dport 22 -j ACCEPT
```

As you can see, the output looks just like the commands that were used to create them, without the preceding `iptables` command. This will also look similar to the iptables rules configuration files, if you've ever used `iptables-persistent` or `iptables save`.

List Specific Chain

If you want to limit the output to a specific chain (`INPUT`, `OUTPUT`, `TCP`, etc.), you can specify the chain name directly after the `-S` option. For example, to show all of the rule specifications in the `TCP` chain, you would run this command:

```
$ sudo iptables -S TCP
```

```
Example: TCP Chain Rule Specification Listing
-N TCP
-A TCP -p tcp -m tcp --dport 22 -j ACCEPT
```

Let's take a look at the alternative way to view the active iptables rules, as a table of rules.

List Rules as Tables

Listing the iptables rules in the table view can be useful for comparing different rules against each other,

To output all of the active iptables rules in a table, run the `iptables` command with the `-L` option:

```
$ sudo iptables -L
```

This will output all of current rules sorted by chain.

If you want to limit the output to a specific chain (`INPUT`, `OUTPUT`, `TCP`, etc.), you can specify the chain name directly after the `-L` option.

Let's take a look at an example INPUT chain:

```
$ sudo iptables -L INPUT
```

```
Example: Input Chain Rule Table Listing
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
UDP        udp  --  anywhere             anywhere             ctstate NEW
TCP        tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN ctstate NEW
ICMP       icmp --  anywhere             anywhere             ctstate NEW
REJECT     udp  --  anywhere             anywhere             reject-with icmp-port-unreachable
REJECT     tcp  --  anywhere             anywhere             reject-with tcp-reset
REJECT     all  --  anywhere             anywhere             reject-with icmp-proto-unreachable
```

The first line of output indicates the chain name (INPUT, in this case), followed by its default policy (DROP). The next line consists of the headers of each column in the table, and is followed by the chain's rules. Let's go over what each header indicates:

- **target**: If a packet matches the rule, the target specifies what should be done with it. For example, a packet can be accepted, dropped, logged, or sent to another chain to be compared against more rules
- **prot**: The protocol, such as `tcp`, `udp`, `icmp`, or `all`
- **opt**: Rarely used, this column indicates IP options
- **source**: The source IP address or subnet of the traffic, or `anywhere`
- **destination**: The destination IP address or subnet of the traffic, or `anywhere`

The last column, which is not labeled, indicates the **options** of a rule. That is, any part of the rule that isn't indicated by the previous columns. This could be anything from source and destination ports, to the connection state of the packet.

Show Packet Counts and Aggregate Size

When listing iptables rules, it is also possible to show the number of packets, and the aggregate size of the packets in bytes, that matched each particular rule. This is often useful when trying to get a rough idea of which rules are matching against packets. To do so, simply use the `-L` and `-v` option together.

For example, let's look at the INPUT chain again, with the `-v` option:

```
$ sudo iptables -L INPUT -v
```

```
Example: Verbose Listing
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
 284K   42M ACCEPT     all  --  any    any     anywhere             anywhere             ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     all  --  lo     any     anywhere             anywhere
    0     0 DROP       all  --  any    any     anywhere             anywhere             ctstate INVALID
  396 63275 UDP        udp  --  any    any     anywhere             anywhere             ctstate NEW
17067 1005K TCP        tcp  --  any    any     anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN cts
 2410  154K ICMP       icmp --  any    any     anywhere             anywhere             ctstate NEW
  396 63275 REJECT     udp  --  any    any     anywhere             anywhere             reject-with icmp-port-unreachable
 2916  179K REJECT     all  --  any    any     anywhere             anywhere             reject-with icmp-proto-unreachabl
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISH
```

Note that the listing now has two additional columns, `pkts` and `bytes`.

Now that you know how to list the active firewall rules in a variety of ways, let's look at how you can reset the packet and byte counters.

Reset Packet Counts and Aggregate Size

If you want to clear, or zero, the packet and byte counters for your rules, use the `-Z` option. They also reset if a reboot occurs. This is useful if you want to see if your server is receiving new traffic that matches your existing rules.

To clear the counters for all chains and rules, use the `-Z` option by itself:

```
$ sudo iptables -Z
```

To clear the counters for all rules in a specific chain, use the `-Z` option and specify the chain. For example, to clear the INPUT chain counters run this command:

```
$ sudo iptables -Z INPUT
```

If you want to clear the counters for a specific rule, specify the chain name and the rule number. For example, to zero the counters for the 1st rule in the INPUT chain, run this:

```
$ sudo iptables -Z INPUT 1
```

Now that you know how to reset the iptables packet and byte counters, let's look at the two methods that can be used to delete them.

Delete Rule by Specification

One of the ways to delete iptables rules is by rule specification. To do so, you can run the `iptables` command with the `-D` option followed by the rule specification. If you want to delete rules using this method, you can use the output of the rules list, `iptables -S`, for some help.

For example, if you want to delete the rule that drops invalid incoming packets (`-A INPUT -m conntrack --ctstate INVALID -j DROP`), you could run this command:

```
$ sudo iptables -D INPUT -m conntrack --ctstate INVALID -j DROP
```

Note that the `-A` option, which is used to indicate the rule position at creation time, should be excluded here.

Delete Rule by Chain and Number

The other way to delete iptables rules is by its **chain** and **line number**. To determine a rule's line number, list the rules in the table format and add the `--line-numbers` option:

```
$ sudo iptables -L --line-numbers
```

```
[secondary_output Example Output: Rules with Line Numbers]
```

```
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
2    ACCEPT     all  --  anywhere             anywhere
3    DROP       all  --  anywhere             anywhere             ctstate INVALID
4    UDP        udp  --  anywhere             anywhere             ctstate NEW
5    TCP        tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN ctstate NEW
6    ICMP       icmp --  anywhere             anywhere             ctstate NEW
7    REJECT     udp  --  anywhere             anywhere             reject-with icmp-port-unreachable
8    REJECT     tcp  --  anywhere             anywhere             reject-with tcp-reset
9    REJECT     all  --  anywhere             anywhere             reject-with icmp-proto-unreachable
10   ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
...
```

This adds the line number to each rule row, indicated by the **num** header.

Once you know which rule you want to delete, note the chain and line number of the rule. Then run the `iptables -D` command followed by the chain and rule number.

For example, if we want to delete the input rule that drops invalid packets, we can see that it's rule `3` of the `INPUT` chain. So we should run this command:

```
$ sudo iptables -D INPUT 3
```

Now that you know how to delete individual firewall rules, let's go over how you can **flush** chains of rules.

Flush Chains

Iptables offers a way to delete all rules in a chain, or **flush** a chain. This section will cover the variety of ways to do this.

Note: Be careful to not lock yourself out of your server, via SSH, by flushing a chain with a default policy of **drop** or **deny**. If you do, you may need to connect to it via the console to fix your access.

Flush a Single Chain

To flush a specific chain, which will delete all of the rules in the chain, you may use the `-F`, or the equivalent `--flush`, option and the name of the chain to flush.

For example, to delete all of the rules in the `INPUT` chain, run this command:

```
$ sudo iptables -F INPUT
```

Flush All Chains

To flush all chains, which will delete all of the firewall rules, you may use the `-F`, or the equivalent `--flush`, option by itself:

```
$ sudo iptables -F
```

Flush All Rules, Delete All Chains, and Accept All

This section will show you how to flush all of your firewall rules, tables, and chains, and allow all network traffic.

Note: This will effectively disable your firewall. You should only follow this section if you want to start over the configuration of your firewall.

First, set the default policies for each of the built-in chains to `ACCEPT`. The main reason to do this is to ensure that you won't be locked out from your server via SSH:

```
$ sudo iptables -P INPUT ACCEPT
$ sudo iptables -P FORWARD ACCEPT
$ sudo iptables -P OUTPUT ACCEPT
```

Then flush the `nat` and `mangle` tables, flush all chains (`-F`), and delete all non-default chains (`-X`):

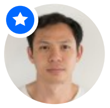
```
$ sudo iptables -t nat -F
$ sudo iptables -t mangle -F
$ sudo iptables -F
$ sudo iptables -X
```

Your firewall will now allow all network traffic. If you list your rules now, you will will see there are none, and only the three default chains (INPUT, FORWARD, and OUTPUT) remain.

Conclusion

After going through this tutorial, you should be familiar with how to list and delete your iptables firewall rules.

Remember that any iptables changes via the `iptables` command are ephemeral, and need to be saved to persist through server reboots. This is covered in the Saving Rules section of the Common Firewall Rules and Commands tutorial.



By Mitchell Anicas

Upvote (34) Subscribe Share

Related Tutorials

How To Migrate Iptables Firewall Rules to a New Server

How To Set Up a Firewall with UFW on Debian 10

Recommended Steps For New FreeBSD 12.0 Servers

How To Secure Nginx with NAXSI on Ubuntu 16.04

How To Set Up a Firewall with UFW on Debian 9

4 Comments

B *I*

Leave a comment...

Log In to Comment

techspecx *September 4, 2016*

0 Another great tutorial I would like to add that is you want to reduce the number of SSH brute force attacks you can add this CHAIN to your iptables rules. Adjust accordingly to the seconds and hitcount and port for your server.

Create chain for ssh attacks

$ iptables -N SSH_CHECK

$ iptables -I INPUT -p tcp --dport 22 -m state --state NEW -j SSH*CHECK*
*$ iptables -A SSH*CHECK -m recent --set --name SSH
$ iptables -A SSH_CHECK -m recent --update --seconds 120 --hitcount 5 --name SSH -j DROP

yash240196 *May 22, 2017*

0 I locked out myself from ssh ? How do i fix it?

0 m9faisal *December 17, 2017*

wow... nice, concise, and useful.. thank you very much

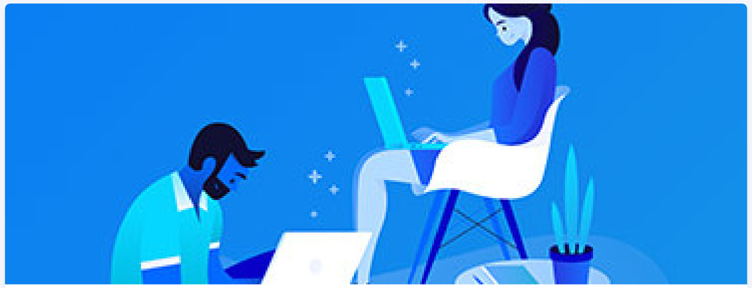
0 Jurijsl *April 12, 2019*

Great tutorial, it worth to add:

iptables -t nat -X

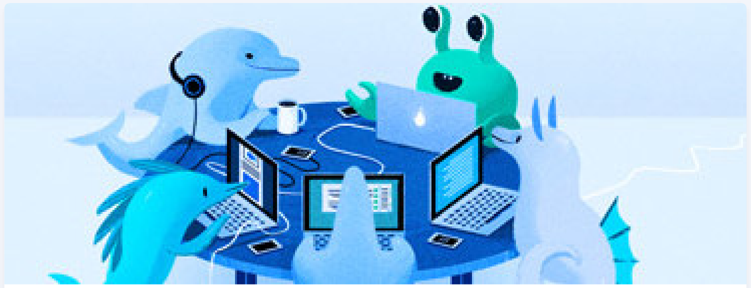
which will clean chains in nat table too.

Thank you!



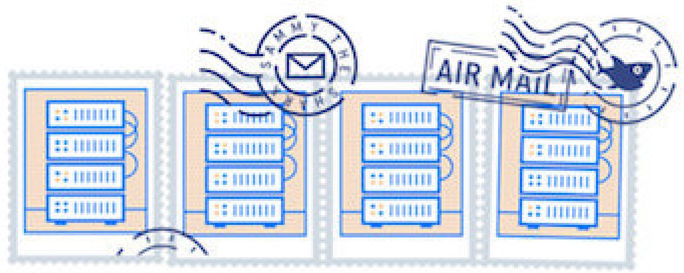
BECOME A CONTRIBUTOR

You get paid; we donate to tech nonprofits.



CONNECT WITH OTHER DEVELOPERS

Find a DigitalOcean Meetup near you.



GET OUR BIWEEKLY NEWSLETTER

Sign up for Infrastructure as a Newsletter.

Welcome to the developer cloud

DigitalOcean makes it simple to launch in the cloud and scale up as you grow – whether you're running one virtual machine or ten thousand.

Learn More

Company

About
Leadership
Blog
Careers
Partners
Referral Program
Press
Legal & Security

Products

Products Overview
Pricing
Droplets
Kubernetes
Managed Databases
Spaces
Marketplace
Load Balancers
Block Storage
Tools & Integrations
API
Documentation
Release Notes

Community

Tutorials
Q&A
Projects and Integrations
Tags
Product Ideas
Meetups
Write for DOnations
Droplets for Demos
Hatch Startup Program
Shop Swag
Research Program
Currents Research
Open Source

Contact

Support
Sales
Report Abuse
System Status