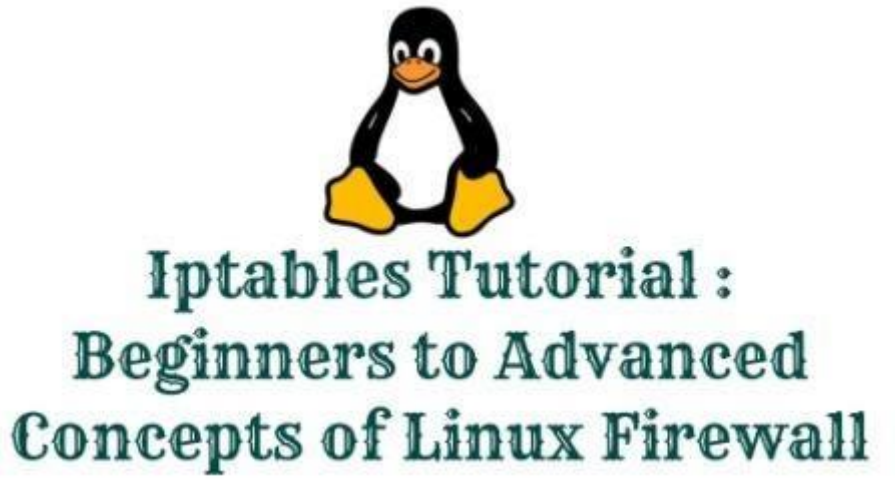
IPTables Tutorial: Beginners to Advanced Guide To Linux Firewall



IPtables is a command-line firewall utility that uses policy chains to allow or block traffic that will be enforced by the linux kernel's netfilter framework. Iptables packet filtering mechanism is organized into three different kinds of structures: **tables**, **chains** and **targets**.

Network traffic is made up of packets. Iptables identifies the packets received and then uses a set of rules to decide what to do with the packets that matches.

Tables and Chains

Iptables has four default tables. Let's have a look on these tables.

1. **Filter**
2. **NAT**
3. **RAW**
4. **Mangle**

1. FILTER

The Filter table is used most frequently. This is default table. It consists of following default chains:

- **INPUT** — This chain handles all packets that are destined to your server and also to control the behaviour for incoming connections. For instance, if a user attempts to SSH into your PC/server, iptables will attempt to match the IP address and port to a rule in the input chain.
- **FORWARD** — This chain is used for packets routed through the system. Think of a router, data is always being sent to it but rarely actually destined for the router itself, the data is just forwarded to its target.

- **OUTPUT** — This chain contains rules for packets generated locally. For instance, if you try to ping google.com, iptables will check its output chain to see what the rules are regarding pinging the google.com.

2. NAT (Network Address Translation)

This table is consulted when a packet tries to create a new connection. It has following chains:

- **PREROUTING** — This chain rule alters a packet as soon as it's received
- **POSTROUTING** — This chain rule alters a packet as it is about to go out.
- **OUTPUT** — This chain rule alerts locally generated traffic.

3. RAW

The Raw table is used to exempt packets from connection tracking. This table consists of two chains:

- **OUTPUT —** To alter locally generated packets
- **PREROUTING —** To alter incoming connections

4. MANGLE

The Mangle table adjusts the IP header properties of packets. The table has all the following chains we described above:

- **INPUT —** for incoming packets
- **OUTPUT —** To alter locally generated packets
- **FORWARD —** for packets routed through the linux box
- **PREROUTING —** To alter incoming connections
- **POSTROUTING —** To alert outgoing connections

Targets

When a packet matches a rule and it is given a **target**, which can be another chain or one of these special options:

- **REJECT** : Means server receives the packet and rejects that packet and also send the acknowledgement.
- **DROP** : Means server receives the packet and drop the request without sending any acknowledgement.
- **ACCEPT** : Means server receives the packet and server allows that request.
- **RETURN** — this rule sends the packet back to the originating chain so you can match it against other rules.

Installing IPtables

Iptables are installed by default on various Linux distributions. To install iptables on debian based linux distributions, you can use below commands:

```
$  sudo apt-get install iptables
```

If you want to keep iptables firewall rules when you reboot the system, then install the persistent package:

```
$  sudo apt-get install iptables-persistent
```

In CentOS 7, iptables was replaced by firewalld. To install iptables, first you need to stop firewalld. The commands stop and prevent firewalld from starting at boot, and mask the service using below command:

```
$  sudo systemctl stop firewalld$  sudo systemctl disable firewalld$  sudo systemctl mask firewalld
```

Next, install and enable iptables on centos 7. First, install the iptables package with the below command:

```
$  sudo yum -y install iptables-services
```

Enter the following commands to enable and start iptables in CentOS 7:

```
$  sudo systemctl enable iptables$  sudo systemctl start iptables$  sudo systemctl status iptables
```

Basic Syntax for iptables Commands & Options

Basic Syntax for iptables is as follows:

```
$  sudo iptables [option] CHAIN_rule [-j target]
```

Here are some common options used in iptables. Iptables is case-sensitive, so make sure you're using the correct options.

- **–A** — append — Add a rule to the chain (at the end)
- **–C** — check — Look for a rule that matches the chain's requirements

- **–D** — delete — Remove specified rules from the chain
- **–F** — flush — Remove all rules.
- **–I** — insert — Add a rule to a chain at some given position.
- **–L** — list — Show all rules in a chain.
- **–N** — new–chain — Create a new chain.
- **–v** — verbose — Show more information when using a list option.
- **–X** — delete–chain — Delete the provided chain.

Configuring iptables Rules

- **Check Current iptables Status**

To get the latest rule status on your server, please enter below command:

```
$  sudo iptables -L# The output of the above commandChain
INPUT (policy ACCEPT)
target    prot opt source              destination
Chain    FORWARD (policy DROP)
target    prot opt source              destination
Chain    OUTPUT (policy ACCEPT)
target    prot opt source                 destination
```

You can get the the status of packets transferred using below command:

```
$  sudo iptables -L -n -v
```

- **Allow loopback Traffic**

You can allow loopback access (access from 127.0.0.1) from localhost. Append the Input chain by entering the following command:

```
$ sudo iptables -A INPUT -i lo -j ACCEPT$ sudo iptables -A OUTPUT -o lo -j ACCEPT
```

The above command configures the firewall to accept traffic for the localhost interface.

- **Block Specific IP Address in IPtables Firewall**

If you find an intrusion activity from an IP address you can block that particular IP address with the below rule:

```
$ iptables -A INPUT -s 192.168.72.128 -j DROP
```

You can block the tcp connections from that particular IP Address using below command.

```
$ iptables -A INPUT -p tcp -s 192.168.72.128 -j DROP
```

You can REJECT traffic from a range of IP addresses, but the command contains multiple options:

```
$ sudo iptables –A INPUT –m iprange –-src-range 192.168.72.1-192.168.72.255 -j REJECT
```

Here **–m** — Match the specified option, **–iprange** — Tell the system to expect a range of IP addresses instead of a single one and **— src-range** — Identifies the range of IP addresses.

- **Unblock IP Address in IPtables Firewall**

Once it is not needed to block that particular IP address, you can unblock it using below command:

```
$  iptables -D INPUT -s 192.168.72.128 -j DROP
```

Using **-D** option you can delete the rule from the chain.

- **Allow Traffic on Specific Ports**

A port is a communication endpoint. Using below commands, you can open ports for HTTP web traffic, HTTPS and SSH (Secure Shell).

```
$  sudo iptables –A INPUT –p tcp  --dport 80 –j ACCEPT$  sudo iptables –A INPUT –p tcp --dport 22 –j ACCEPT$  sudo iptables –A INPUT –p tcp --dport 443 –j ACCEPT
```

Here **-p** (protocol) checks for specified port whether tcp or udp, **–dport** specifies for destination port and **-j** (jump) takes the specified action.

- **Block Specific Port on IPtables Firewall**

To block outgoing connections on a specific port, you can use below command:

```
$  iptables -A OUTPUT -p tcp --dport 80 -j DROP
```

- **Allow Multiple Ports on IPtables using Multiport**

There will be multiple occasions you have to allow multiple ports at once then you can use below options:

```
$  iptables -A INPUT -p tcp -m multiport --dport  22,80,443 -j ACCEPT$  iptables -A OUTPUT -p tcp -m multiport --sport 22,80,443 -j ACCEPT
```

Here **–m** — Match the specified option, **multiport** — Tell the system to expect a range of IP addresses instead of a single one, **—dport** — destination port and **–sport** — source port.

- **Allow Specific Network Range on Particular Port on IPtables**

The below command can be used to open network range on specific port.

```
$  iptables -A OUTPUT -p tcp -d 192.168.72.128/24 --dport 22 -j ACCEPT
```

- **Block Incoming Ping Requests on IPtables**

You can block the ping request from open network.

```
$  iptables -A INPUT -p icmp -i eth0 -j DROP
```

- **Flush IPtables Firewall Chains or Rules**

If you want to delete all tables then following command can be used:

```
$  sudo iptables -F
```

You can flush chains from specific table with:

```
$  iptables -t nat -F
```

- **Save IPtables Rules to a File**

It’s up to you where you want to store the file with any name:

```
$  iptables-save > ~/iptables.rules
```

- **Restore IPtables Rules from a File**

The saved rule file can be restored using following command:

```
$  iptables-restore < ~/iptables.rules
```

- **Dropping Unwanted Traffic**

If you already define dport firewall rules, you need to prevent unauthorized access by dropping any traffic that comes through other ports:

```
$  sudo iptables –A INPUT –j DROP
```

The –A option appends a new rule to the chain. If any connection comes through ports other than those you defined, it will be dropped.

- **Delete a Rule**

You can use the –F option to clear all iptables firewall rules. A more precise method is to delete the line number of a rule.

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

Check the line number of the firewall rule you want to delete and run below command:

```
$  sudo iptables –D INPUT <Number>
```

Put the number of actual rule line number you want to remove.