Course Code : CSL604 **Course Name : System Security Lab**

Class : TE-CO **Batch : Computer Engineering**

Roll no : 18CO63 **Name : SHAIKH TAUSEEF MUSHTAQUE ALI**

Experiment : 08

Aim : Setting up personal Firewall using iptables.

Theory :

iptables is a firewall program for **Linux**. It will monitor traffic from and to your server using **tables**. These tables contain **sets of rules**, called **chains**, that will filter incoming and outgoing data packets.

[optin-monster-shortcode id="fv4lqeko3gylvecpszws"]

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

- **ACCEPT** – will allow the packet to pass through.
- **DROP** – will not let the packet pass through.
- **RETURN** – stops the packet from traversing through a chain and tell it to go back to the previous chain.

In this iptables tutorial, we are going to work with one of the default tables, called **filter**. It consists of three chains:

- **INPUT** – controls incoming packets to the server.
- **FORWARD** – filters incoming packets that will be forwarded somewhere else.
- **OUTPUT** – filter packets that are going out from your server.

Before we begin this guide, make sure you have SSH **root** or **sudo** access to your machine that runs on Ubuntu 16.04 or up. You can establish the connection through **PuTTY** (Windows) or terminal shell (Linux, macOS). If you own Hostinger VPS, you can get the SSH login details on the Servers tab of hPanel.

iptables rules only apply to **ipv4**. If you want to set up a firewall for the **ipv6** protocol, you will need to use **ip6tables** instead.

How to Install and Use Iptables Linux Firewall

We will divide this iptables tutorial into three steps. First, you will learn how to install the tool on Ubuntu. Secondly, we are going to show you how to define the rules. Lastly, we will guide you to make persistent changes in iptables.

Step 1 — Installing Iptables

Iptables comes pre-installed in most Linux distributions. However, if you don't have it in Ubuntu/Debian system by default, follow the steps below:

1. Connect to your server via SSH. If you don't know, you can read our **SSH tutorial**.
2. Execute the following command one by one:

sudo apt-get update

sudo apt-get install iptables

3. Check the status of your current iptables configuration by running:

sudo iptables -L -v

4. Here, the -L option is used to list all the rules, and -v is for showing the info in a more detailed format. Below is the example output:

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)

pkts bytes target prot opt in out source destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)

pkts bytes target prot opt in out source destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)

pkts bytes target prot opt in out source destination

You will now have the Linux firewall installed. At this point, you can notice that all chains are set to **ACCEPT** and have no rules. This is not secure since any packet can come through without filtering.

Don't worry. We'll tell you how to define rules on the next step of our iptables tutorial.

Step 2 – Defining Chain Rules

Defining a rule means appending it to the chain. To do this, you need to insert the **-A** option (**Append**) right after the iptables command, like so:
sudo iptables -A

It will alert iptables that you are adding new rules to a chain. Then, you can combine the command with other options, such as:

- **-i** (**interface**) — the network interface whose traffic you want to filter, such as eth0, lo, ppp0, etc.
- **-p** (**protocol**) — the network protocol where your filtering process takes place. It can be either **tcp**, **udp**, **udplite**, **icmp**, **sctp**, **icmpv6**, and so on. Alternatively, you can type **all** to choose every protocol.
- **-s** (**source**) — the address from which traffic comes from. You can add a hostname or IP address.
- **–dport** (**destination port**) — the destination port number of a protocol, such as **22** (**SSH**), **443** (**https**), etc.
- **-j** (**target**) — the target name (**ACCEPT**, **DROP**, **RETURN**). You need to insert this every time you make a new rule.

If you want to use all of them, you must write the command in this order:
sudo iptables -A <chain> -i <interface> -p <protocol (tcp/udp) > -s <source> --dport <port no.> -j <target>

Once you understand the basic syntax, you can start configuring the firewall to give more security to your server. For this iptables tutorial, we are going to use the **INPUT** chain as an example.

Enabling Traffic on Localhost

To allow traffic on localhost, type this command:
sudo iptables -A INPUT -i lo -j ACCEPT

For this iptables tutorial, we use **lo** or **loopback** interface. It is utilized for all communications on the localhost. The command above will make sure that the connections between a database and a web application on the same machine are working properly.

Enabling Connections on HTTP, SSH, and SSL Port

Next, we want **http** (port **80**), **https** (port **443**), and **ssh** (port **22**) connections to work as usual. To do this, we need to specify the protocol (**-p**) and the corresponding port (**–dport**). You can execute these commands one by one:

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT

sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT

sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

It's time to check if the rules have been appended in iptables:

```
sudo iptables -L -v
```

It should return with the results below which means all TCP protocol connections from the specified ports will be accepted:

```
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:ssh
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:http
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:https
```

Filtering Packets Based on Source

Iptables allows you to filter packets based on an IP address or a range of IP addresses. You need to specify it after the **-s** option. For example, to accept packets from **192.168.1.3**, the command would be:

```
sudo iptables -A INPUT -s 192.168.1.3 -j ACCEPT
```

You can also reject packets from a specific IP address by replacing the **ACCEPT** target with **DROP**.

```
sudo iptables -A INPUT -s 192.168.1.3 -j DROP
```

If you want to drop packets from a range of IP addresses, you have to use the **-m** option and **iprange** module. Then, specify the IP address range with **–src-range**. Remember, a hyphen should separate the range of ip addresses without space, like this:

```
sudo iptables -A INPUT -m iprange --src-range 192.168.1.100-192.168.1.200 -j DROP
```

Dropping all Other Traffic

It is crucial to use the **DROP** target for all other traffic after defining **–dport** rules. This will prevent an unauthorized connection from accessing the server via other open ports. To achieve this, simply type:
sudo iptables -A INPUT -j DROP

Now, the connection outside the specified port will be dropped.

Deleting Rules

If you want to remove all rules and start with a clean slate, you can use the **-F** option (**flush**):
sudo iptables -F

This command erases all current rules. However, to delete a specific rule, you must use the -D option. First, you need to see all the available rules by entering the following command:
sudo iptables -L --line-numbers

You will get a list of rules with numbers:
Chain INPUT (policy ACCEPT)

num target prot opt source destination

1 ACCEPT all -- 192.168.0.4 anywhere

2 ACCEPT tcp -- anywhere anywhere tcp dpt:https

3 ACCEPT tcp -- anywhere anywhere tcp dpt:http

4 ACCEPT tcp -- anywhere anywhere tcp dpt:ssh

To delete a rule, insert the corresponding chain and the number from the list. Let's say for this iptables tutorial, we want to get rid of **rule number three** of the **INPUT** chain. The command should be:
sudo iptables -D INPUT 3

Step 3 – Persisting Changes

The iptables rules that we have created are saved in memory. That means we have to redefine them on reboot. To make these changes persistent after restarting the server, you can use this command:
sudo /sbin/iptables-save

It will save the current rules on the system configuration file, which will be used to reconfigure the tables every time the server reboots.

Note that you should always run this command every time you make changes to the rules. For example, if you want to disable iptables, you need to execute these two lines:

```
sudo iptables -F

sudo /sbin/iptables-save
```

You will see the following results:

```
root@vps42707999:~# sudo /sbin/iptables-save
# Generated by iptables-save v1.6.1 on Tue Nov 19 11:38:18 2019
*nat
:PREROUTING ACCEPT [645426:37524813]
:INPUT ACCEPT [558190:28836818]
:OUTPUT ACCEPT [263056:18809892]
:POSTROUTING ACCEPT [263056:18809892]
:OUTPUT_direct - [0:0]
:POSTROUTING_ZONES - [0:0]
:POSTROUTING_ZONES_SOURCE - [0:0]
:POSTROUTING_direct - [0:0]
:POST_public - [0:0]
:POST_public_allow - [0:0]
```

Conclusion:

Personal Firewall using iptables was set up.