Name: Robert Acosta

Instructor: Adrianna Holden-Gouveia

Subject: Linux Administration

Date: 11 November 2020

# Firewalls

Iptables : Iptables is the main command that allows Linux users to modify firewall rules and network traffic.

**Definitions of iptables commands**

**-A** : Add or append a rule to the INPUT chain.

**-I** : Insert one or more rules in the head of the selected chain.

**-p** : To indicate the protocol (TCP, UMP, ICM)

**-s** : To indicate the source of the packet

**-d** : To indicate the destination of the packet

**-j** : Indicates what should happen to the packet

-j ACCEPT : Accepts the packet

**-o** : Indicates which outgoing packets are sent through the Input, Forward and PREROUTING.

**INPUT** : Controls the behavior of the incoming connections.

**FORWARD** : Rerouted incoming connections that is forwarded to a specific target

**OUTPUT** : Controls the behavior of outgoing connections.

**--dport** : destination port

**--sport** : source port

**-s** : source

**-D** : Delete rule

**Iptables Manual**

iptables manual currently have 3 types of tables:

**FILTER** – this is the default table, which contains the built-in chains for:

INPUT  – packages destined for local sockets

FORWARD – packets routed through the system

OUTPUT – packets generated locally

**NAT** – a table that is consulted when a packet tries to create a new connection. It has the following built-in:

PREROUTING – used for altering a packet as soon as it is received

OUTPUT – used for altering locally generated packets

POSTROUTING – used for altering packets as they are about to go out

**MANGLE** – this table is used for packet altering.

PREROUTING – for altering incoming connections

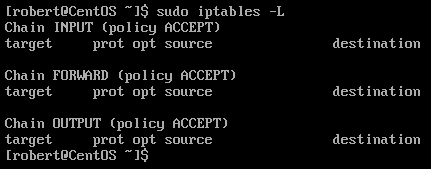
OUTPUT – for altering locally generated  packets

INPUT – for incoming packets

POSTROUTING – for altering packets as they are about to go out

FORWARD – for packets routed through the box

**sudo iptables -L**  : List your currently firewall rules



As we can see in the picture there are no rules added in the iptables firewall.

1. Deal with web server (open needed ports, and forward port 80 traffic to 8080)

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

Make sure to run the command in administrator mode with “sudo”. -A means to add a rule. INPUT to the incoming connection. -p stands for protocol, I selected the tcp protocol. --dport stands for the destination port, I selected port 80. -j What should happen to the packet ? ACCEPT which means to accept or allow.

Let us do the same thing for port 8080

**sudo iptables -A INPUT -p tcp --dport 8080 -j ACCEPT**

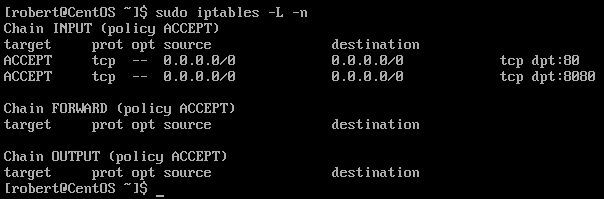
To Forward port 80 traffic to 800

**sudo iptables -A PREROUTING -t nat -p tcp --dport 80 -j REDIRECT --to-port 8080**

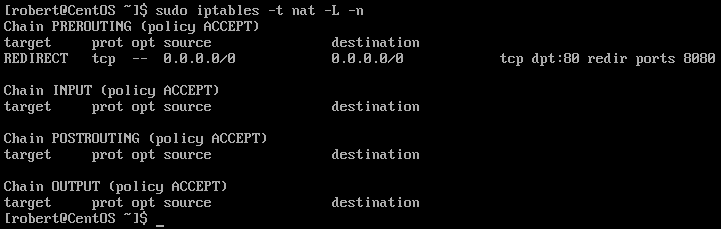
Iptables -A to add a new rule to the iptables. PREROUTING to alter the incoming packet before reaching the destination (Port 80). -t nat is used to extract the RREROUTING function from the NAT table. Remember that iptables does not have the PREROUTING function-built in. Then we select the protocol -p tcp and the destination port --dport 80. It must be the same as the one that we opened in the example above. -j What should happen to the packet? We use the function REDIRECT self-explanatory; it will redirect the traffic. --to-port , to what port ? 8080

Results:

**Iptables -L -n** : List your currently rules and display the ports

 As we can see in the picture above port 80 and port 8080 are open.

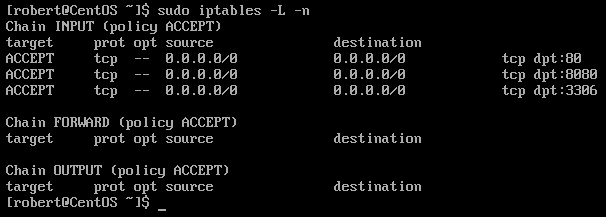
**Iptables -t nat -L -n** : List your currently rules from the NAT iptables and is display the ports

 As we can see in the picture above port 80 redirect its traffic to port 8080.

1. Deal with MySQL service (open needed ports)

MySQL as a default use port 3306 and TCP as its main protocol.

**sudo iptables -A INPUT -p tcp --dport 3306 -j ACCEPT**



1. Deal with SSH service (allow incoming and outgoing SSH, second script to deny SSH)

SSH, Secure Shell uses TCP port 22.

To allow incoming traffic from SSH :

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

To allow outgoing traffic from SSH :

**sudo iptables -A OUTPUT -p tcp --dport 22 -j ACCEPT**

OUPUT is used for outgoing traffic

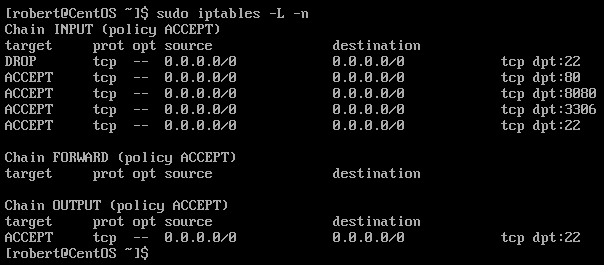
To block all SSH traffic :

**sudo iptables -I INPUT -p tcp --dport 22 -j DROP**

DROP is used to drop all traffics. -I is used to insert a new rule to an existing rule. Reason I did not use -A append is because I do not want to overwrite an existing rule.

**sudo iptables -A INPUT -p tcp -s 123.123.123 -dport 22 -j ACCEPT**

This command allows us to whitelist a specific IP. It is basically the same as the other commands but in this case, we add the function -s which mean source. The port 22 will allow incoming connection from the IP 123.123.123.



1. Deal with Email service (Such as allow or block incoming SMTP,POP3...)

There is not a single SMTP port. Instead, there are four STMP ports. They are port 25, 587, 465, 2525.

To allow incoming traffic from SMTP :

**sudo iptables -A INPUT -p tcp --dport 25 -j ACCEPT**

**sudo iptables -A INPUT -p tcp --dport 587 -j ACCEPT**

**sudo iptables -A INPUT -p tcp --dport 465 -j ACCEPT**

**sudo iptables -A INPUT -p tcp --dport 2525 -j ACCEPT**

To block incoming traffic from SMTP **:**

**sudo iptables -A INPUT -p tcp --dport 25 -j DROP**

**sudo iptables -A INPUT -p tcp --dport 587 -j DROP**

**sudo iptables -A INPUT -p tcp --dport 465 -j DROP**

**sudo iptables -A INPUT -p tcp --dport 2525 -j DROP**

POP3 works on two ports: port 143 and port 993

To allow incoming traffic from POP3 :

**sudo iptables -A INPUT -p tcp --dport 143 -j ACCEPT**

**sudo iptables -A INPUT -p tcp --dport 993 -j ACCEPT**

To block incoming traffic from POP3 :

**sudo iptables -A INPUT -p tcp --dport 143 -j DROP**

**sudo iptables -A INPUT -p tcp --dport 993 -j DROP**

1. Script to allow/block specific hosts, MAC addresses, block telnet, block ping

To block a specific host from incoming connection:

**sudo iptables -A INPUT -s 123.123.123 DROP**

To allow incoming connection from a specific host :

**sudo iptables -A INPUT -s 123.123.125 ACCEPT**

To allow specific MAC addresses :

**iptables -A INPUT -m mac --mac-source 00:0F:EA:91:04:08 -j DROP**

To use the MAC address function, we must type : -m mac and --mac-source to specify the MAC source.

To allow specific MAC addresses :

**iptables -A INPUT -m mac --mac-source 00:0F:EA:91:04:08 -j ACCEPT**

Note : INPUT for incoming connection OUTPUT for outgoing connection.

Telnet uses port 23 as is default port

How to allow incoming telnet communication :

**sudo iptables -A INPUT -p tcp --dport 23 -j ACCEPT**

How to block incoming telnet communication :

**sudo iptables -A INPUT -p tcp --dport 23 -j DROP**

How to disable ping request :

**sudo iptables -A INPUT -p --icmp-type echo-request -j REJECT**

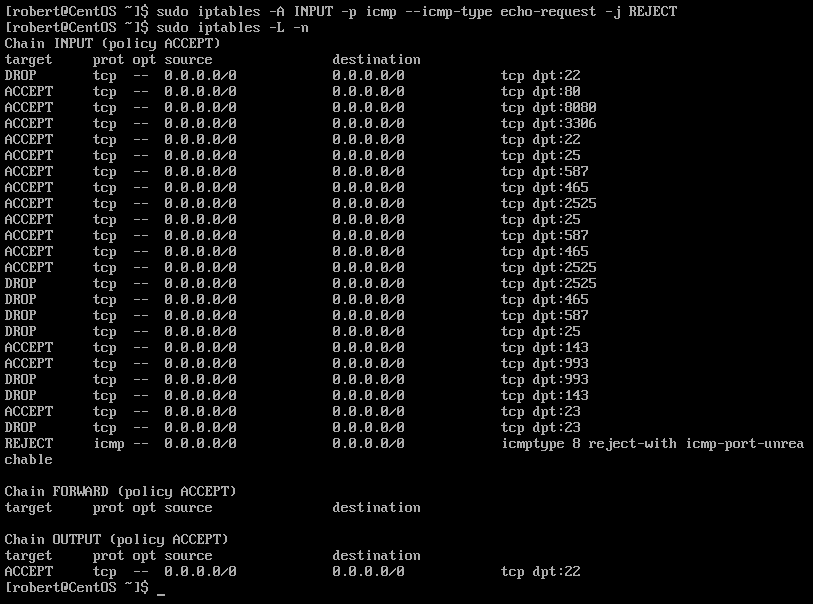
Note: In networking ping request are ICMP packet request.

How to enable back ping request:

**iptables -D INPUT -p icmp --icmp-type echo-request -j REJECT**

**-D** : This function will delete the existing rule

## My existing Rules

****

1. Write a script to backup your Firewall/iptables

How to backup your existing Firewall rules:

**sudo iptables-save > filename.backup**

How to restore firewall rules from a backup file

**sudo iptables-restore < filename.backup**

1. Use Wireshark to show that your statements/scripts are working properly.

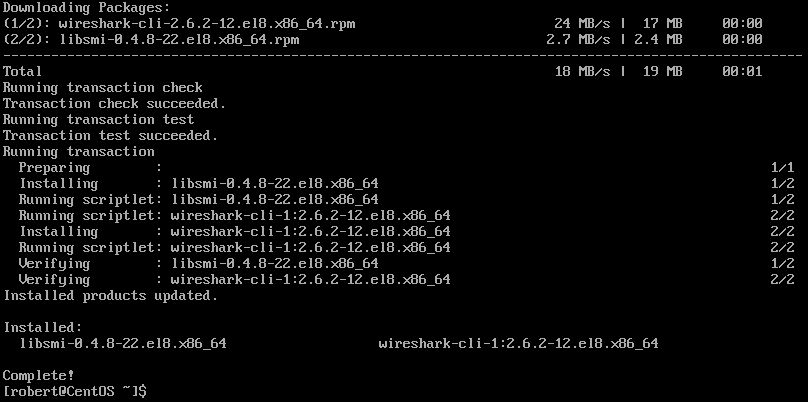
We must install Wireshark on the server first. If we do not have a GUI/Desktop in our server, we must install the Wireshark command line tool.

**sudo dnf install wireshark-cli** : This is the command to install the Wireshark command line tool.

Note: dnf is a package manager similar to “YUM”.

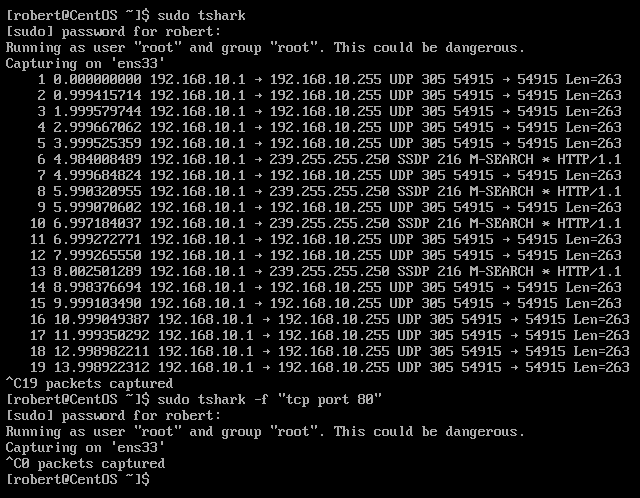


In the keyboard type “y” as yes to download the package.



The installation is complete. Now we can use Wireshark to monitor the server network.

**tshark** : command to run the Wireshark command line tool.



1. Write the specs for how you think you could prevent a DDOS attack using IPtables.  Can you write a script for this?

I will limit the amount of ICMP echo request as well as limit the amount of connection per sources.

**#!/bin/bash**

**/sbin/iptables -A INPUT -p icmp -m limit --limit 2/second --limit-burst 2 -j ACCEPT**

**/sbin/iptables -A INPUT -p tcp -m connlimit --connlimit-above 50 -j REJECT --reject-with tcp-reset**

I would set a rule to stop smurfs attacks

**/sbin/iptables ­A INPUT ­p icmp ­m icmp ­­icmp­type address­mask­request ­j DROP**

**/sbin/iptables ­A INPUT ­p icmp ­m icmp ­­icmp­type timestamp­request ­j DROP**

**/sbin/iptables ­A INPUT ­p icmp ­m icmp ­j DROP**

I will set a rule to Drop all invalid packets

**/sbin/iptables ­A INPUT ­m state ­­state INVALID ­j DROP**

**/sbin/iptables ­A FORWARD ­m state ­­state INVALID ­j DROP**

**/sbin/iptables ­A OUTPUT ­m state ­­state INVALID ­j DROP**

References

Hosting, SiteGround Web. “Email Protocols - POP3, SMTP and IMAP Tutorial.” *SiteGround Knowledge Resources*, www.siteground.com/tutorials/email/protocols-pop3-smtp-imap/.

“How to Choose the Right SMTP Port (Port 25, 587, 465, or 2525).” *Kinsta Managed WordPress Hosting*, 14 Feb. 2020, kinsta.com/blog/smtp-port/.

Iftekher, Mohammad Forhad, et al. “IPTABLES VS FIREWALLD.” *Unixmen*, 4 May 2016, www.unixmen.com/iptables-vs-firewalld/.

Mitesh Soni - Priyanka Agashe -, et al. “Iptables: The Default Linux Firewall.” *Open Source For You*, 9 June 2016, www.opensourceforu.com/2015/04/iptables-the-default-linux-firewall/.

Todorov, Marin, et al. “Marin Todorov.” *Tecmint*, 1 Mar. 2016, www.tecmint.com/linux-iptables-firewall-rules-examples-commands/.