Experiment 7 :- To control network traffic using iptables rules.

Name: Vedant Sandip Kannurkar
UID: 2024801005
Division: B2

Exercise 1: iptables Essentials:

Steps performed :

View all rules in the filter table : iptables -L

```
admin4@gordon-freeman:~$ sudo iptables -L
[sudo] password for admin4:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination
DOCKER-USER  all  --  anywhere             anywhere
DOCKER-ISOLATION-STAGE-1  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
DOCKER     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain DOCKER (1 references)
target     prot opt source               destination

Chain DOCKER-ISOLATION-STAGE-1 (1 references)
target     prot opt source               destination
DOCKER-ISOLATION-STAGE-2  all  --  anywhere             anywhere
RETURN     all  --  anywhere             anywhere

Chain DOCKER-ISOLATION-STAGE-2 (1 references)
target     prot opt source               destination
DROP       all  --  anywhere             anywhere
RETURN     all  --  anywhere             anywhere

Chain DOCKER-USER (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
```

Verbose view : iptables -L -v

Experiment 7 :- To control network traffic using iptables rules.

```
admin4@gordon-freeman:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DOCKER-USER  all  --  any    any     anywhere             anywhere
    0     0 DOCKER-ISOLATION-STAGE-1  all  --  any    any     anywhere             anywhere
    0     0 ACCEPT     all  --  any    docker0  anywhere             anywhere             ctst
    0     0 DOCKER     all  --  any    docker0  anywhere             anywhere
    0     0 ACCEPT     all  --  docker0 !docker0  anywhere             anywhere
    0     0 ACCEPT     all  --  docker0 docker0  anywhere             anywhere

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain DOCKER (1 references)
 pkts bytes target     prot opt in     out     source               destination

Chain DOCKER-ISOLATION-STAGE-1 (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DOCKER-ISOLATION-STAGE-2  all  --  docker0 !docker0  anywhere             anywhere
    0     0 RETURN     all  --  any    any     anywhere             anywhere

Chain DOCKER-ISOLATION-STAGE-2 (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  any    docker0  anywhere             anywhere
    0     0 RETURN     all  --  any    any     anywhere             anywhere

Chain DOCKER-USER (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 RETURN     all  --  any    any     anywhere             anywhere
```

To display only the rules under the INPUT chain:

```
admin4@gordon-freeman:~$ sudo iptables -v -L INPUT
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

To display all the rules under the mangle table:

```
admin4@gordon-freeman:~$ sudo iptables -L -t mangle
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

To display all the rules under the nat table:

Experiment 7 :- To control network traffic using iptables rules.

```
admin4@gordon-freeman:~$ sudo iptables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DOCKER     all  --  anywhere             anywhere             ADDRTYPE match dst-type LOCAL

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DOCKER     all  --  anywhere            !localhost/8          ADDRTYPE match dst-type LOCAL

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  172.17.0.0/16        anywhere

Chain DOCKER (2 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
```

To flush all the current rules:

```
admin4@gordon-freeman:~$ sudo iptables --flush
```

To create your own chain:

```
admin4@gordon-freeman:~$ sudo iptables -N mychain
admin4@gordon-freeman:~$ sudo iptables -L mychain
Chain mychain (0 references)
target     prot opt source               destination
```

To delete chains

```
admin4@gordon-freeman:~$ sudo iptables -X INPUT
iptables: No chain/target/match by that name.
admin4@gordon-freeman:~$ sudo iptables -X mychain
admin4@gordon-freeman:~$ sudo iptables -L mychain
iptables v1.8.7 (nf_tables): chain `mychain' in table `filter' is incompatible, use 'nft' tool.
```

Exercise 2: Basic Packet Filtering

To filter ICMP packets types: Before starting ensure that you can ping your partner system and that your partner system can ping you too successfully.

Experiment 7 :- To control network traffic using iptables rules.

```
admin4@gordon-freeman:~$ sudo ping -c 2 serverPR
ping: serverPR: Temporary failure in name resolution
admin4@gordon-freeman:~$ sudo ping -c 2 www.google.com
PING www.google.com (142.251.220.4) 56(84) bytes of data.
64 bytes from pnbomb-ay-in-f4.1e100.net (142.251.220.4): icmp_seq=1 ttl=117 time=1.63 ms
64 bytes from pnbomb-ay-in-f4.1e100.net (142.251.220.4): icmp_seq=2 ttl=117 time=10.2 ms

--- www.google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 5074ms
rtt min/avg/max/mdev = 1.633/5.913/10.194/4.280 ms
```

Flush all your existing rules

```
admin4@gordon-freeman:~$ sudo iptables -F
```

Create a rule to prevent all outgoing icmp type packets to any destination. Test it using ping

```
admin4@gordon-freeman:~$ sudo iptables -A OUTPUT -o enp2s0 -p icmp -j DROP
admin4@gordon-freeman:~$ sudo ping -c 2 serverPR
ping: serverPR: Temporary failure in name resolution
admin4@gordon-freeman:~$ sudo ping -c 2 www.google.com
PING www.google.com (142.251.220.4) 56(84) bytes of data.
^C
--- www.google.com ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1056ms
```

View the rule you just created.

```
admin4@gordon-freeman:~$ sudo iptables -vL OUTPUT
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2   168 DROP       icmp --  any    enp2s0  anywhere             anywhere
```

Flush all the rules and execute the ping command again.

```
admin4@gordon-freeman:~$ sudo iptables -F
admin4@gordon-freeman:~$ sudo ping -c 2 www.google.com
PING www.google.com (142.251.220.4) 56(84) bytes of data.
64 bytes from pnbomb-ay-in-f4.1e100.net (142.251.220.4): icmp_seq=1 ttl=117 time=1.99 ms
64 bytes from pnbomb-ay-in-f4.1e100.net (142.251.220.4): icmp_seq=2 ttl=117 time=2.02 ms

--- www.google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.989/2.003/2.017/0.014 ms
```

Now create another rule that will drop ICMP packets that originate from a specific unwanted IP address

```
admin4@gordon-freeman:~$ sudo iptables -A INPUT -i enp2s0 -p icmp --source 172.16.0.44 -j DROP
```

Experiment 7 :- To control network traffic using iptables rules.

Instead of flushing all the rules in your tables. Delete only the rule you created above. To do this you need to know the rule number. Using the line-number that matches the rule in you want to delete, you can delete the specific rule

```
admin4@gordon-freeman:~$ sudo iptables -vL INPUT --line-numbers
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 DROP       icmp --  enp2s0 any     172.16.0.44          anywhere
admin4@gordon-freeman:~$ sudo iptables -D INPUT 1
```

To filter other kinds of traffic, restart ftp

```
admin4@gordon-freeman:~$ sudo service vsftpd start
admin4@gordon-freeman:~$ sudo service vsftpd restart
```

Block FTP from partner

```
admin4@gordon-freeman:~$ sudo iptables -A INPUT -i enp2s0 -s www.google.com -p tcp --dport 21 -j DROP
admin4@gordon-freeman:~$ sudo iptables -A INPUT -s www.google.com -p tcp --dport 80 -j DROP
```

Questions

1. What option is needed to get a more verbose version of this command iptables -L -t nat?

Ans : iptables -L -t nat -v

2. What is the command to display the rules under the OUTPUT chain?

Answer : iptables -L OUTPUT

3. What port does the ftp service “normally” listen on?

Answer : Port 21

4. What is the command to create a chain called “mynat-chain” under the nat table?

Experiment 7 :- To control network traffic using iptables rules.

Answer : iptables -t nat -N mynat-chain

5. Research online and list the names of some easier to use tools or applications that can be used to manage the firewall sub-system on Linux based systems.

Answer :

● UFW (Uncomplicated Firewall) – user-friendly frontend for iptables

● Firewalld – dynamic firewall manager with zones

● GUFW – graphical interface for UFW

● Shorewall (Shoreline Firewall) – high-level tool for configuring iptables

● CSF (ConfigServer Security & Firewall) – advanced firewall UI for servers

---

Conclusion

In this experiment, I explored the functionality and practical application of iptables for managing network traffic and enhancing system security. By configuring various rules and chains, I learned how to control incoming and outgoing packets, implement firewall policies, and prevent unauthorized access. The hands-on experience helped reinforce theoretical knowledge of networking and security, and highlighted the importance of rule order and chain priority in effective firewall configuration.