PA3 : IPTABLES

Keshav Chhabra(2022247)

1.

Part a)

Configuring for the **client** network adapter using yaml file in /etc/netplan

```
network:
   version: 2
   renderer: networkd
   ethernets:
      enp0s8:
        addresses: [20.1.1.1/24]
        routes:
          - to: default
            via: 20.1.1.2
```

The changes are reflected on the interface enp0s8 now with the IP address : 20.1.1.1

```
root@client:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN group default qlen 1000
    link/ether 08:00:27:04:da:83 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:cf:5c:64 brd ff:ff:ff:ff:ff:ff
    inet 20.1.1.1/24 brd 20.1.1.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fecf:5c64/64 scope link
       valid_lft forever preferred_lft forever
```

Similarly , configuring for the **gateway's** network adapters 1 (20.1.1.2) and network adapter 2(40.1.1.1)

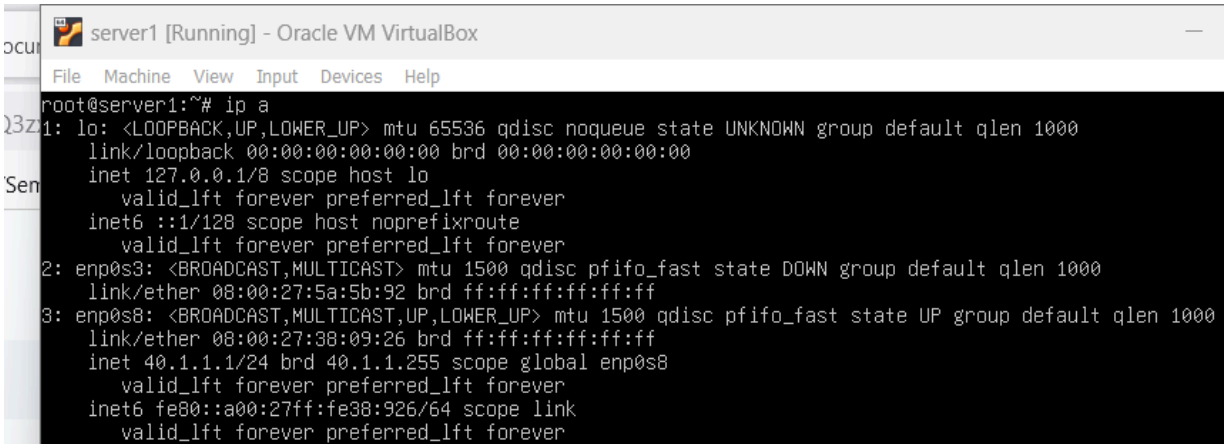
```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s8:
      addresses: [20.1.1.2/24]
      dhcp4: no
```

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s9:
      addresses: [40.1.1.2/24]
      dhcp4: no
```

```
root@gateway:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:50:c5:5c brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 82046sec preferred_lft 82046sec
    inet6 fe80::a00:27ff:fe50:c55c/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:7a:b7:bb brd ff:ff:ff:ff:ff:ff
    inet 20.1.1.2/24 brd 20.1.1.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7a:b7bb/64 scope link
       valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:27:e8:e0 brd ff:ff:ff:ff:ff:ff
    inet 40.1.1.2/24 brd 40.1.1.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe27:e8e0/64 scope link
       valid_lft forever preferred_lft forever
```

Configuring for **server1** with IP address 40.1.1.1

```
network:
  version: 2
  renderer: networkd
  ethernets:
     enp0s8:
        addresses: [40.1.1.1/24]
        dhcp4: no
        gateway4: 40.1.1.2
```



Configuring for **server2** with IP address 40.1.1.3

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s8:
      addresses: [40.1.1.3/24]
      dhcp4: no
      gateway4: 40.1.1.2
```

```
root@server2:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN group default qlen 1000
    link/ether 08:00:27:01:37:69 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:7a:80:ad brd ff:ff:ff:ff:ff:ff
    inet 40.1.1.3/24 brd 40.1.1.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7a:80ad/64 scope link
       valid_lft forever preferred_lft forever
```

Part b) Configuring VM2 as the gateway such that it can **forward** the incoming traffic to one of the servers .

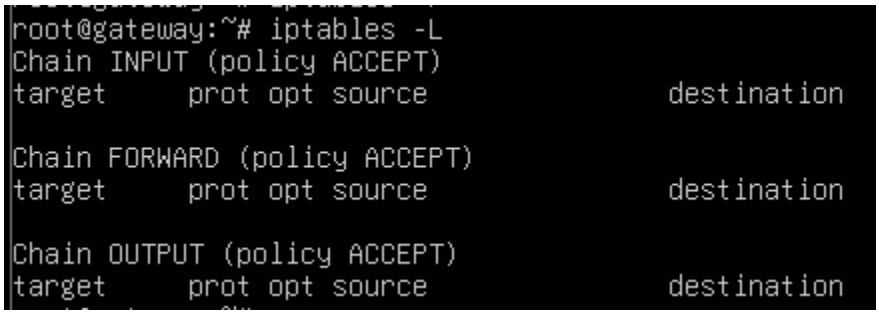
```
root@gateway:~# sudo sysctl -w net.ipv4.ip_forward=1_
```

The **systcl** command is used to inspect and modify kernel parameters that control various system behaviors. The -w option allows users to write a specified value to a kernel parameter. For example, net.ipv4.ip_forward=1 modifies the IPv4 settings, specifically enabling IP

forwarding by setting the ip_forward parameter to 1. Conversely, setting it to 0 would disable IP forwarding. This functionality is essential for configuring network routing in a Linux environment.

2.

Part a) Initially no rules are mentioned in the iptables at gateway.



This command ensures that any request of ping(an icmp echo message) to 40.1.1.1 gets forwarded to the destination

The following command **sudo iptables -A FORWARD -d 40.1.1.1 -j DROP** ensures that all other packets to 40.1.1.1 that come to the gateway are dropped . Since the iptable's rules are processed in a chronological order , the ping message would get accepted before it could be dropped .

```
root@gateway:~# sudo iptables -A FORWARD -p icmp --icmp-type 8 -d 40.1.1.1 -j ACCEPT
root@gateway:~# sudo  iptables -A FORWARD -d 40.1.1.1 -j DROP
root@gateway:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             40.1.1.1             icmp echo-request
DROP       all  --  anywhere             40.1.1.1

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

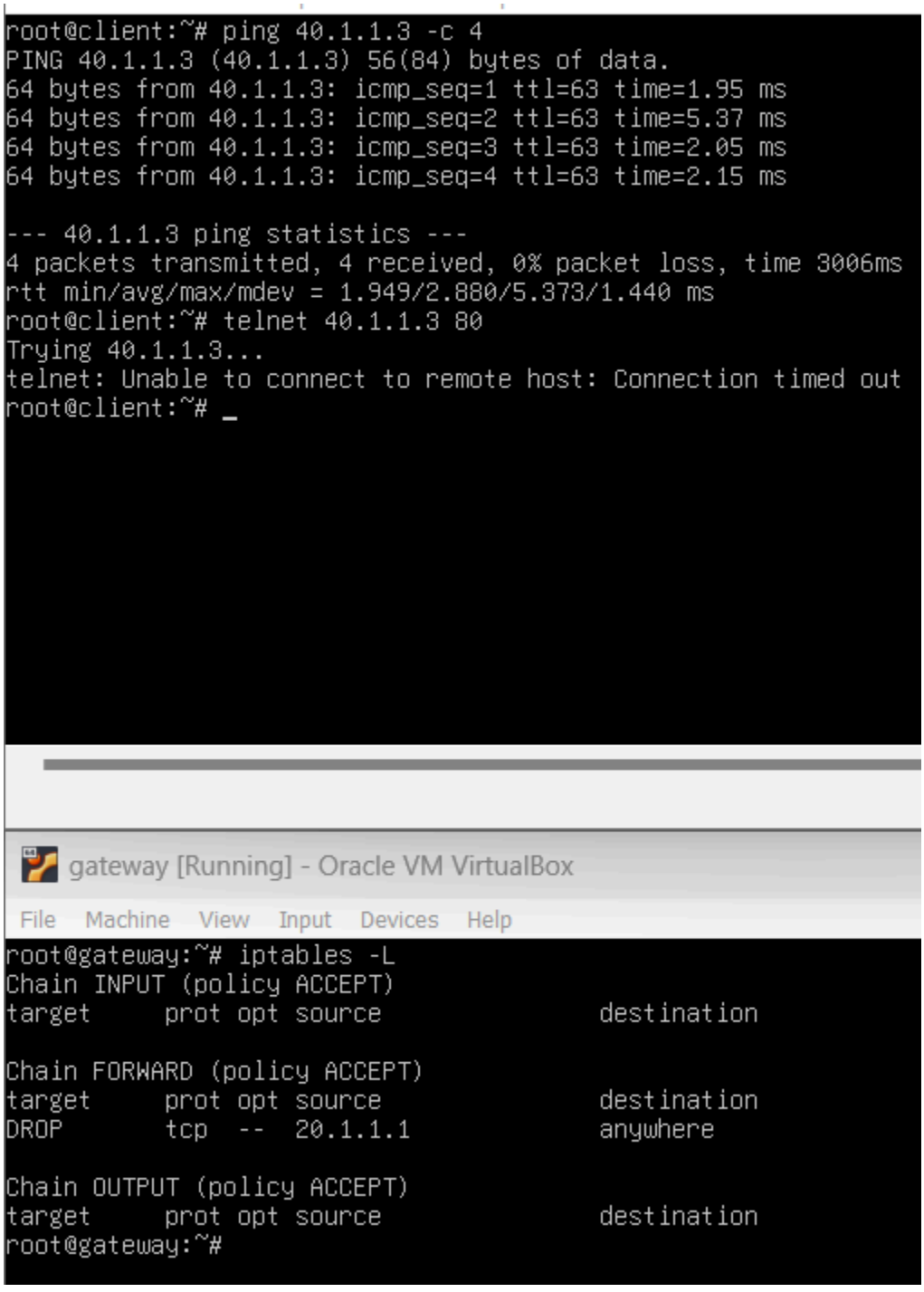
After this we try running **telnet** but as it is not a ping message , icmp of echo request type , it is dropped .

```
root@client:~# telnet 40.1.1.1

Trying 40.1.1.1...
telnet: Unable to connect to remote host: Connection timed out
root@client:~#
```

On the other hand , **ping** gets forwarded as accepted

```
root@client:~# ping 40.1.1.1 -c 4
PING 40.1.1.1 (40.1.1.1) 56(84) bytes of data.
64 bytes from 40.1.1.1: icmp_seq=1 ttl=63 time=52.0 ms
64 bytes from 40.1.1.1: icmp_seq=2 ttl=63 time=59.9 ms
64 bytes from 40.1.1.1: icmp_seq=3 ttl=63 time=29.4 ms
64 bytes from 40.1.1.1: icmp_seq=4 ttl=63 time=64.6 ms

--- 40.1.1.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3669ms
rtt min/avg/max/mdev = 29.422/51.476/64.609/13.512 ms
```

Part b)
In this , we are dropping any TCP request initiated by 20.1.1.1 as can be seen using iptables -L
We show this by executing telnet , which runs on tcp which gets blocked , whereas ping which runs on icmp , gets forwarded .



The **TCP** connection from source 20.1.1.1 will be dropped at the gateway so there is no connection transaction that occurs .

```
root@client:~# iperf -c 40.1.1.3
------------------------------------------------------------
Client connecting to 40.1.1.3, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
```

```
root@gateway:~# sudo iptables -F
root@gateway:~# sudo iptables -A FORWARD -d 40.1.1.1 -p icmp -j ACCEPT
root@gateway:~# sudo iptables -A FORWARD -d 40.1.1.1 -j DROP
root@gateway:~# sudo iptables -A FORWARD -s 20.1.1.1 -p tcp -j DROP
root@gateway:~# ``_
```

```
root@server-2:~# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
_
```

For **UDP** , we pass additional -u as a flag to denote that we want to check for UDP connection using the iperf tool .

```
root@client:~# iperf -c 40.1.1.3 -u
------------------------------------------------------------
Client connecting to 40.1.1.3, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 20.1.1.1 port 54084 connected with 40.1.1.3 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0154 sec  1.25 MBytes  1.05 Mbits/sec
[  1] Sent 896 datagrams
[  1] Server Report:
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  1] 0.0000-10.0151 sec  1.25 MBytes  1.05 Mbits/sec   0.111 ms 0/895 (0%)
root@client:~# `
```

We see a client connecting to IP address 40.1.1.3 on UDP port 5001, sending 1470-byte datagrams with a target interval of around 11215 microseconds

```
root@gateway:~# sudo iptables -F
root@gateway:~# sudo iptables -A FORWARD -d 40.1.1.1 -p icmp -j ACCEPT
root@gateway:~# sudo iptables -A FORWARD -d 40.1.1.1 -j DROP
root@gateway:~# sudo iptables -A FORWARD -s 20.1.1.1 -p tcp -j DROP
root@gateway:~#
```

This section displays iptables firewall rules being configured on a gateway system, specifically setting up FORWARD chain rules to ACCEPT and DROP traffic between the 40.1.1.1 and 20.1.1.1 networks.

```
root@server-2:~# iperf -u -s
------------------------------------------------------------
Server listening on UDP port 5001
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 40.1.1.3 port 5001 connected with 20.1.1.1 port 54084
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  1] 0.0000-10.0151 sec  1.25 MBytes  1.05 Mbits/sec   0.111 ms 0/895 (0%)
root@server-2:~#
```

This portion reveals an Iperf server listening on UDP port 5001, showing the results of the connection test.

The test results indicate a bandwidth of approximately 1.05 Mbits/sec with zero packet loss over a 10-second interval, and a jitter of 0.111 ms. The consistent bandwidth and zero packet loss suggest a stable network connection between the client and server .

Part b)

i) Here we ping 40.1.1.1 from the client virtual machine 20.1.1.1

```
--- 40.1.1.1 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19027ms
rtt min/avg/max/mdev = 1.659/2.306/2.970/0.362 ms
```

ii) The following are the stats for 40.1.1.3 when pinged from 20.1.1.1

```
--- 40.1.1.3 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19027ms
rtt min/avg/max/mdev = 1.807/2.174/2.822/0.265 ms
```

iii)

Based on the above observations , the performance measures of rtt : min / avg / max /mdev are approximately similar . This is because no configurations are changed and both server1 and server2 are behaving symmetrically for the client .

4.

Part a)

Initially there are no routing rules in the iptables at the gateway .

```
root@gateway:~# sudo iptables -t nat -L POSTROUTING -v -n --line-numbers
Chain POSTROUTING (policy ACCEPT 101 packets, 9822 bytes)
num   pkts bytes target     prot opt in     out     source               destination
```

```
root@client:~# ping -c 5 40.1.1.1
PING 40.1.1.1 (40.1.1.1) 56(84) bytes of data.
64 bytes from 40.1.1.1: icmp_seq=1 ttl=63 time=2.63 ms
64 bytes from 40.1.1.1: icmp_seq=2 ttl=63 time=2.87 ms
64 bytes from 40.1.1.1: icmp_seq=3 ttl=63 time=2.12 ms
64 bytes from 40.1.1.1: icmp_seq=4 ttl=63 time=1.89 ms
64 bytes from 40.1.1.1: icmp_seq=5 ttl=63 time=2.12 ms

--- 40.1.1.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 1.887/2.324/2.872/0.365 ms
root@client:~# _
```

Here we can see that upon pinging server 1 from 20.1.1.1 it shows the source address for echo request as **20.1.1.1** which is the **client** server .

```
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:30:45.025538 IP 20.1.1.1 > server1: ICMP echo request, id 1090, seq 3, length 64
10:30:45.025562 IP server1 > 20.1.1.1: ICMP echo reply, id 1090, seq 3, length 64
10:30:46.027598 IP 20.1.1.1 > server1: ICMP echo request, id 1090, seq 4, length 64
10:30:46.027621 IP server1 > 20.1.1.1: ICMP echo reply, id 1090, seq 4, length 64
10:30:47.028807 IP 20.1.1.1 > server1: ICMP echo request, id 1090, seq 5, length 64
10:30:47.028833 IP server1 > 20.1.1.1: ICMP echo reply, id 1090, seq 5, length 64
10:30:48.187641 ARP, Request who-has _gateway tell server1, length 28
10:30:48.188377 ARP, Reply _gateway is-at 08:00:27:27:e8:e0 (oui Unknown), length 46
10:30:48.257858 ARP, Request who-has server1 tell _gateway, length 46
```

The following command helps us perform the operation of changing the source IP address of every packet from 20.1.1.1/24 to 40.1.1.2/24

```
root@gateway:~# sudo iptables -t nat -A POSTROUTING -s 20.1.1.1 -j SNAT --to-source 40.1.1.2
root@gateway:~# sudo iptables -t nat -L POSTROUTING -v -n --line-numbers
Chain POSTROUTING (policy ACCEPT 101 packets, 9822 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 SNAT       0    --  *      *       20.1.1.1             0.0.0.0/0            to:40.1.1.2
```

After making these changes , when we ping from 20.1.1.1 , it shows the source address is _gateway instead of the client (20.1.1.1) , indicating that desirable changes have been made .

Part b)
This is tcpdump at the interface enp0s9 with IP address **40.1.1.1** on the gateway VM . Here we can see that introducing the rule in postrouting table as above translates the src address of the echo request packet from client to _gateway , as desired by part a)
More importantly , when the reply comes , it has as its destination host _gateway and we are required to **retranslate** it to the destination client .

```
10:36:12.348537 ARP, Reply server1 is-at 08:00:27:38:09:26 (oui Unknown), length 28
10:36:13.007070 IP _gateway > server1: ICMP echo request, id 1099, seq 7, length 64
10:36:13.007097 IP server1 > _gateway: ICMP echo reply, id 1099, seq 7, length 64
10:36:14.008447 IP _gateway > server1: ICMP echo request, id 1099, seq 8, length 64
10:36:14.008485 IP server1 > _gateway: ICMP echo reply, id 1099, seq 8, length 64
10:36:15.010521 IP _gateway > server1: ICMP echo request, id 1099, seq 9, length 64
10:36:15.010544 IP server1 > _gateway: ICMP echo reply, id 1099, seq 9, length 64
10:36:16.011531 IP _gateway > server1: ICMP echo request, id 1099, seq 10, length 64
10:36:16.011555 IP server1 > _gateway: ICMP echo reply, id 1099, seq 10, length 64
10:36:16.549619 IP 192.168.56.1 > igmp.mcast.net: igmp v3 report, 1 group record(s)
```

The command given below will help us achieve the required objective for this part .
It matches packets destined for 40.1.1.2
Performs destination NAT, changing the destination IP address of the matched packets to 20.1.1.1 .

```
root@gateway:~# iptables -t nat -A PREROUTING -d 40.1.1.2   -j DNAT --to-destination 20.1.1.1
```

```
tcpdump -i enp0s8
```

The following is the tcpdump at the client side at interface enp0s8 with IPaddress 20.1.1.1.
It shows that it is sending echo request packets , and after making the changes in the routing table , we can see that the echo reply are coming back to the client (20.1.1.1) even though we saw at adapter interface 2 (**IP: 40.1.1.2**) that the packets were destined for the _gateway .

```
11:30:08.378498 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 15, length 64
11:30:09.378981 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 16, length 64
11:30:09.380915 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 16, length 64
11:30:10.380005 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 17, length 64
11:30:10.382075 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 17, length 64
11:30:11.380894 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 18, length 64
11:30:11.383015 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 18, length 64
11:30:12.382921 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 19, length 64
11:30:12.384524 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 19, length 64
11:30:13.384923 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 20, length 64
11:30:13.387054 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 20, length 64
11:30:14.385959 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 21, length 64
11:30:14.387544 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 21, length 64
11:30:15.388413 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 22, length 64
11:30:15.390378 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 22, length 64
11:30:16.388884 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 23, length 64
11:30:16.391308 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 23, length 64
11:30:17.390070 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 24, length 64
11:30:17.392278 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 24, length 64
11:30:18.391323 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 25, length 64
11:30:18.393305 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 25, length 64
11:30:19.391886 IP client > 40.1.1.1: ICMP echo request, id 1219, seq 26, length 64
11:30:19.394163 IP 40.1.1.1 > client: ICMP echo reply, id 1219, seq 26, length 64
```

This is **tcpdump** at the server1 .

```
11:30:27.369362 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 34, length 64
11:30:28.370365 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 35, length 64
11:30:28.371437 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 35, length 64
11:30:29.371537 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 36, length 64
11:30:29.372447 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 36, length 64
11:30:30.373438 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 37, length 64
11:30:30.374413 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 37, length 64
11:30:31.374381 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 38, length 64
11:30:31.375406 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 38, length 64
11:30:32.375386 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 39, length 64
11:30:32.376455 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 39, length 64
11:30:33.377467 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 40, length 64
11:30:33.378353 IP 40.1.1.1 > gateway: ICMP echo reply, id 1219, seq 40, length 64
11:30:33.655394 ARP, Request who-has 40.1.1.1 tell gateway, length 28
11:30:33.656497 ARP, Reply 40.1.1.1 is-at 08:00:27:38:09:26 (oui Unknown), length 46
11:30:34.378617 IP gateway > 40.1.1.1: ICMP echo request, id 1219, seq 41, length 64
```

5.

Using the information from Q3 part c) , we can infer that the server2 with IP: 40.1.1.3 has lower RTT than server 1 . Hence we will assign a probability = 0.8 to a packet coming to the gateway to be forwarded to this server .

Below are the changes in the rules made in the iptables .

```
root@gateway:~# iptables -t nat -A PREROUTING -d 20.1.1.1 -m statistic --mode random --probability 0.8 -j DNAT --to-destination 40.1.1.3
root@gateway:~#
root@gateway:~#
root@gateway:~#
root@gateway:~#
root@gateway:~# iptables -t nat -A PREROUTING -d 20.1.1.1 -m statistic --mode random --probability 0.2 -j DNAT --to-destination 40.1.1.1
```

server1 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:37:02.211379 IP 20.1.1.1 > server1: ICMP echo request, id 1829, seq 1, length 64
14:37:09.673536 IP 20.1.1.1 > server1: ICMP echo request, id 1843, seq 1, length 64
14:37:24.633788 IP 20.1.1.1 > server1: ICMP echo request, id 1871, seq 1, length 64
14:37:34.248788 IP 20.1.1.1 > server1: ICMP echo request, id 1889, seq 1, length 64
14:37:36.380718 IP 20.1.1.1 > server1: ICMP echo request, id 1893, seq 1, length 64
14:37:41.696613 IP 20.1.1.1 > server1: ICMP echo request, id 1903, seq 1, length 64
14:37:48.098572 IP 20.1.1.1 > server1: ICMP echo request, id 1915, seq 1, length 64
14:37:52.369698 IP 20.1.1.1 > server1: ICMP echo request, id 1923, seq 1, length 64
14:37:59.854530 IP 20.1.1.1 > server1: ICMP echo request, id 1937, seq 1, length 64
14:38:05.194620 IP 20.1.1.1 > server1: ICMP echo request, id 1947, seq 1, length 64
14:38:06.263621 IP 20.1.1.1 > server1: ICMP echo request, id 1949, seq 1, length 64
14:38:08.396541 IP 20.1.1.1 > server1: ICMP echo request, id 1953, seq 1, length 64
14:38:11.561556 IP 20.1.1.1 > server1: ICMP echo request, id 1959, seq 1, length 64
14:38:12.634416 IP 20.1.1.1 > server1: ICMP echo request, id 1961, seq 1, length 64
14:38:13.703608 IP 20.1.1.1 > server1: ICMP echo request, id 1963, seq 1, length 64
14:38:17.979653 IP 20.1.1.1 > server1: ICMP echo request, id 1971, seq 1, length 64
14:38:21.181584 IP 20.1.1.1 > server1: ICMP echo request, id 1977, seq 1, length 64
14:38:22.245957 IP 20.1.1.1 > server1: ICMP echo request, id 1979, seq 1, length 64
14:38:28.663579 IP 20.1.1.1 > server1: ICMP echo request, id 1991, seq 1, length 64
^C
19 packets captured
19 packets received by filter
0 packets dropped by kernel
root@server1:~#
```

server2 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
14:38:01.797516 IP 20.1.1.1 > server2: ICMP echo request, id 1941, seq 1, length 64
14:38:02.867258 IP 20.1.1.1 > server2: ICMP echo request, id 1943, seq 1, length 64
14:38:03.936034 IP 20.1.1.1 > server2: ICMP echo request, id 1945, seq 1, length 64
14:38:07.143103 IP 20.1.1.1 > server2: ICMP echo request, id 1951, seq 1, length 64
14:38:09.239346 IP 20.1.1.1 > server2: ICMP echo request, id 1955, seq 1, length 64
14:38:10.306267 IP 20.1.1.1 > server2: ICMP echo request, id 1957, seq 1, length 64
14:38:14.584872 IP 20.1.1.1 > server2: ICMP echo request, id 1965, seq 1, length 64
14:38:15.655136 IP 20.1.1.1 > server2: ICMP echo request, id 1967, seq 1, length 64
14:38:16.723086 IP 20.1.1.1 > server2: ICMP echo request, id 1969, seq 1, length 64
14:38:18.862898 IP 20.1.1.1 > server2: ICMP echo request, id 1973, seq 1, length 64
14:38:19.929239 IP 20.1.1.1 > server2: ICMP echo request, id 1975, seq 1, length 64
14:38:23.128369 IP 20.1.1.1 > server2: ICMP echo request, id 1981, seq 1, length 64
14:38:24.196366 IP 20.1.1.1 > server2: ICMP echo request, id 1983, seq 1, length 64
14:38:25.267359 IP 20.1.1.1 > server2: ICMP echo request, id 1985, seq 1, length 64
14:38:26.333339 IP 20.1.1.1 > server2: ICMP echo request, id 1987, seq 1, length 64
14:38:27.402902 IP 20.1.1.1 > server2: ICMP echo request, id 1989, seq 1, length 64
14:38:29.544827 IP 20.1.1.1 > server2: ICMP echo request, id 1993, seq 1, length 64
14:38:30.612309 IP 20.1.1.1 > server2: ICMP echo request, id 1995, seq 1, length 64
14:38:31.681150 IP 20.1.1.1 > server2: ICMP echo request, id 1997, seq 1, length 64
14:38:32.749822 IP 20.1.1.1 > server2: ICMP echo request, id 1999, seq 1, length 64
14:38:33.813445 IP 20.1.1.1 > server2: ICMP echo request, id 2001, seq 1, length 64
14:38:34.881503 IP 20.1.1.1 > server2: ICMP echo request, id 2003, seq 1, length 64
14:38:35.919854 IP 20.1.1.1 > server2: ICMP echo request, id 2005, seq 1, length 64
^C
81 packets captured
81 packets received by filter
0 packets dropped by kernel
root@server2:~#
```

Part b)
In the above picture , server 1 catches 19 packets while server 2 catches 81 packets , which roughly attributes a probability of 0.8 to the server with lower RTT(server 2) as required .

```
for i in {1..100}; do ping -c 1 40.1.1.1; sleep 1 ; done
```

The above command is used to initiate 100 sequential pings which will be load balanced by the gateway according to rules aforementioned in the iptables of the gateway and caught by server 1 and server 2 with the desired probabilistic load .