INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

acm India Council

ACM India Winter School on “Full-stack Networking”

Day 1 Hands-on session on Linux Networking

Sumit Kumar, Research Scholar, IIIT Delhi
Najiya Naj, Research Scholar, IIIT Delhi
Maryam Tahira, Research Scholar, IIIT Delhi
Rinku Shah, Assistant Professor, IIIT Delhi

Brief Bio

Sumit Kumar
Research Scholar, IIITD
sumitk@iiitd.ac.in

Najiya Naj
Research Scholar, IIITD
najiyan@iiitd.ac.in

Maryam Tahira
Research Scholar,IIITD
maryamt@iiitd.ac.in

Rinku Shah
Assistant Professor, IIITD
rinku@iiitd.ac.in
https://faculty.iiitd.ac.in/~rinku/

Demo Session 1

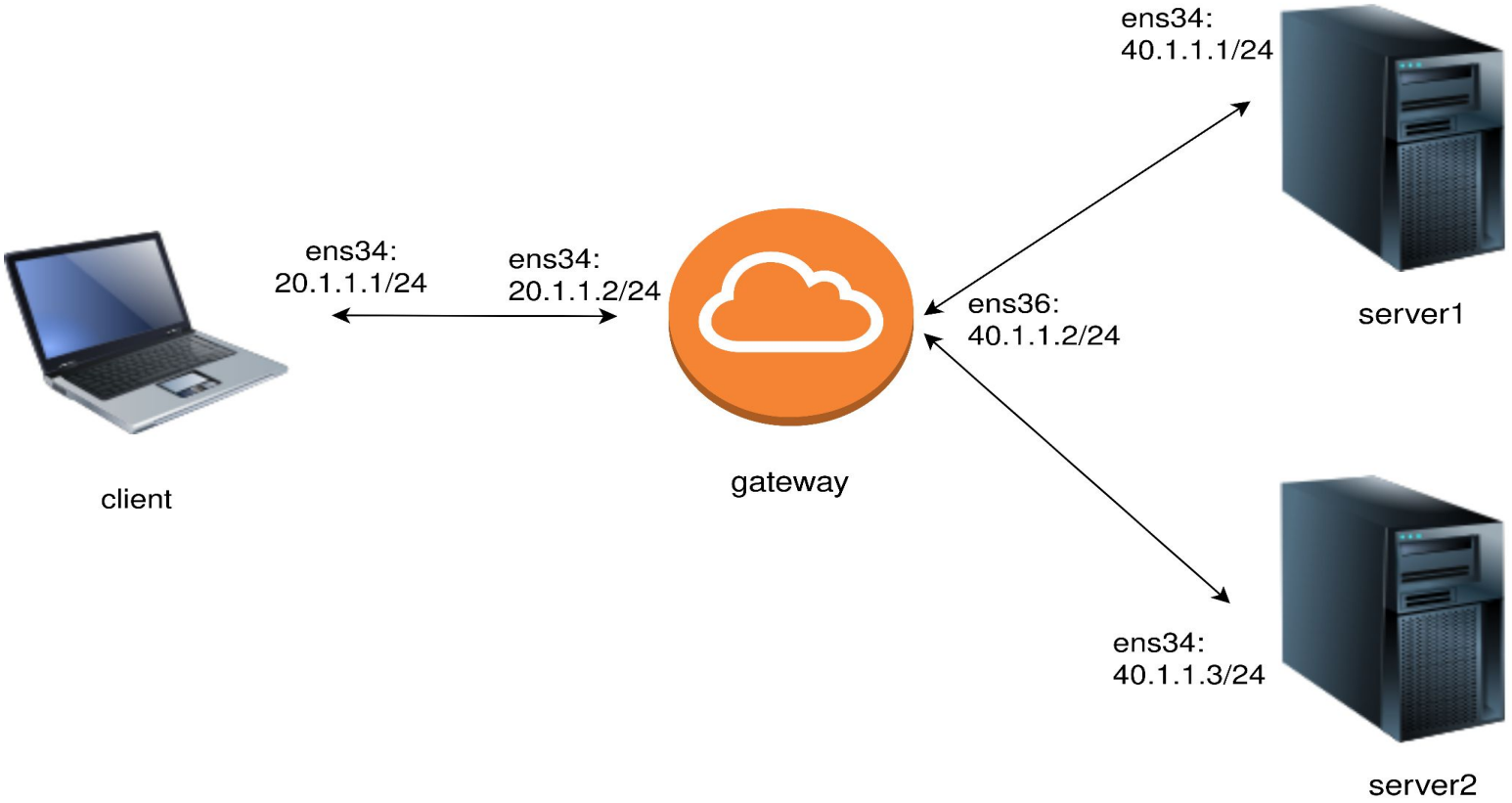
Communication between client and server

- ifconfig, ping, arp, traceroute
- route
- ip forwarding
- tshark

Measurement of throughput between client and server

- Iperf
- netstate

Setup Diagram

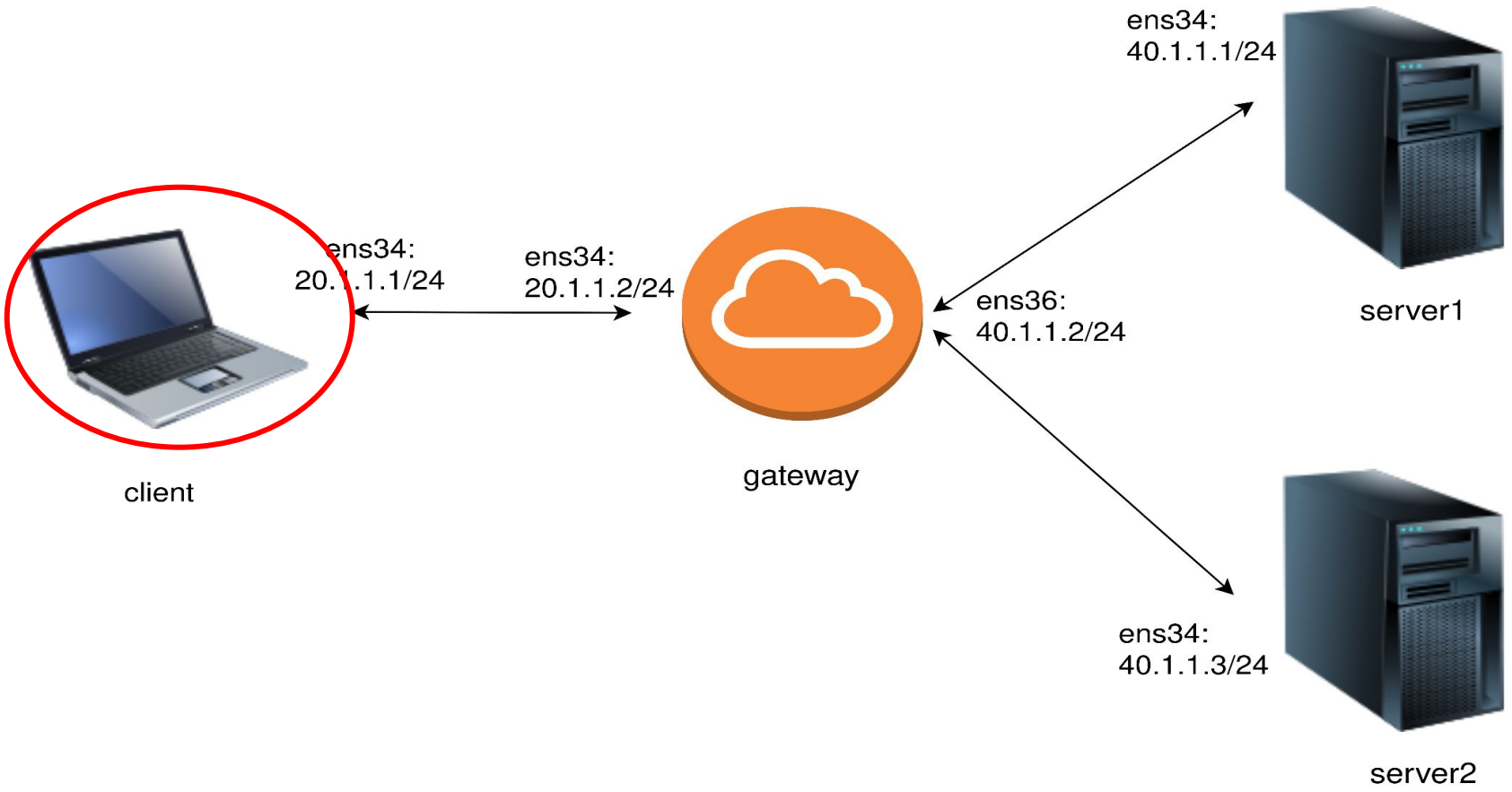


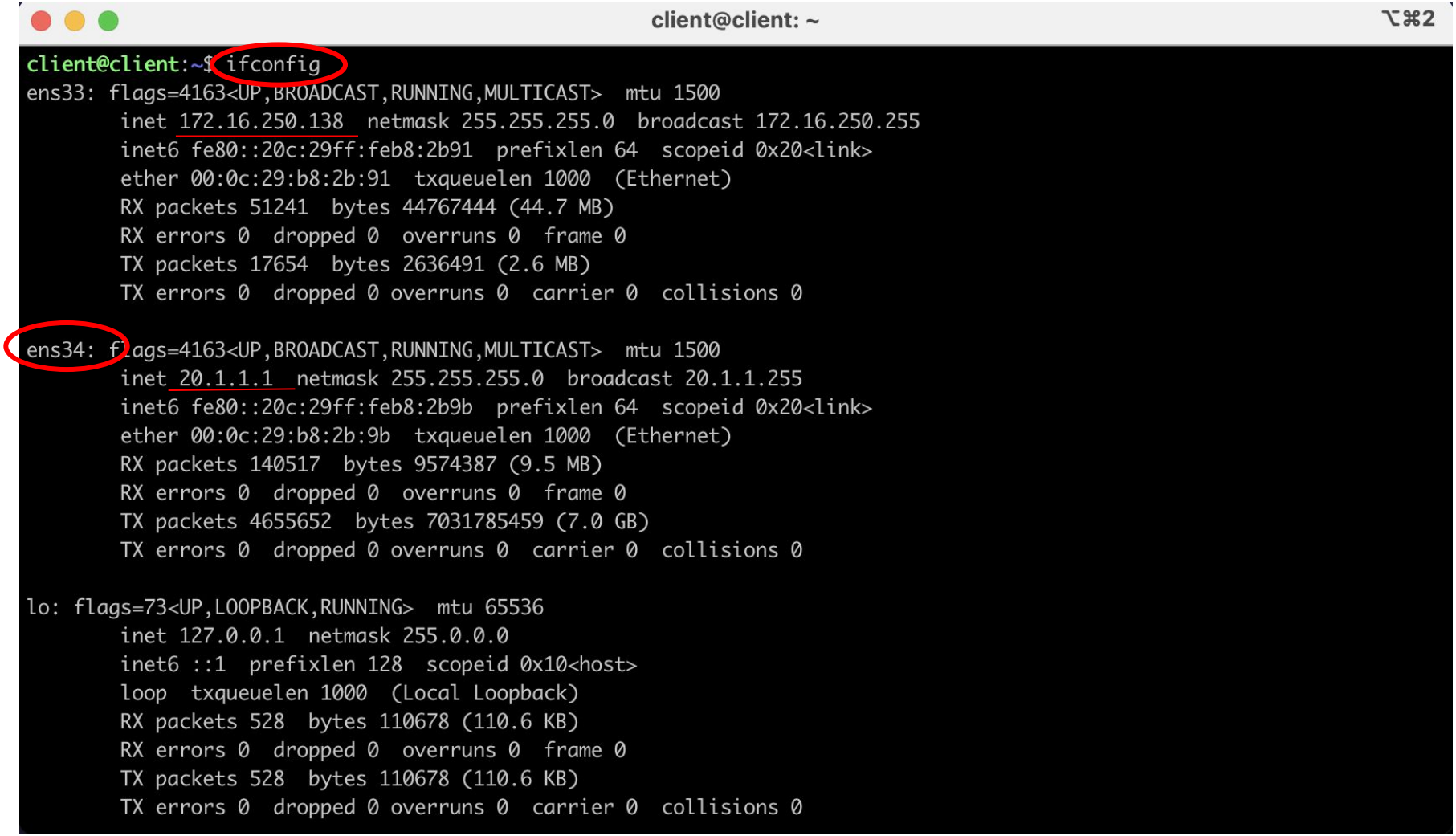
Communication between client and server

ifconfig

- Configures the network interfaces on Unix-like operating systems.
- Enables users to view and modify network interface settings
- ifconfig : display the status of the currently active interface.
- -a : display the status of all interfaces.

```
ifconfig
```

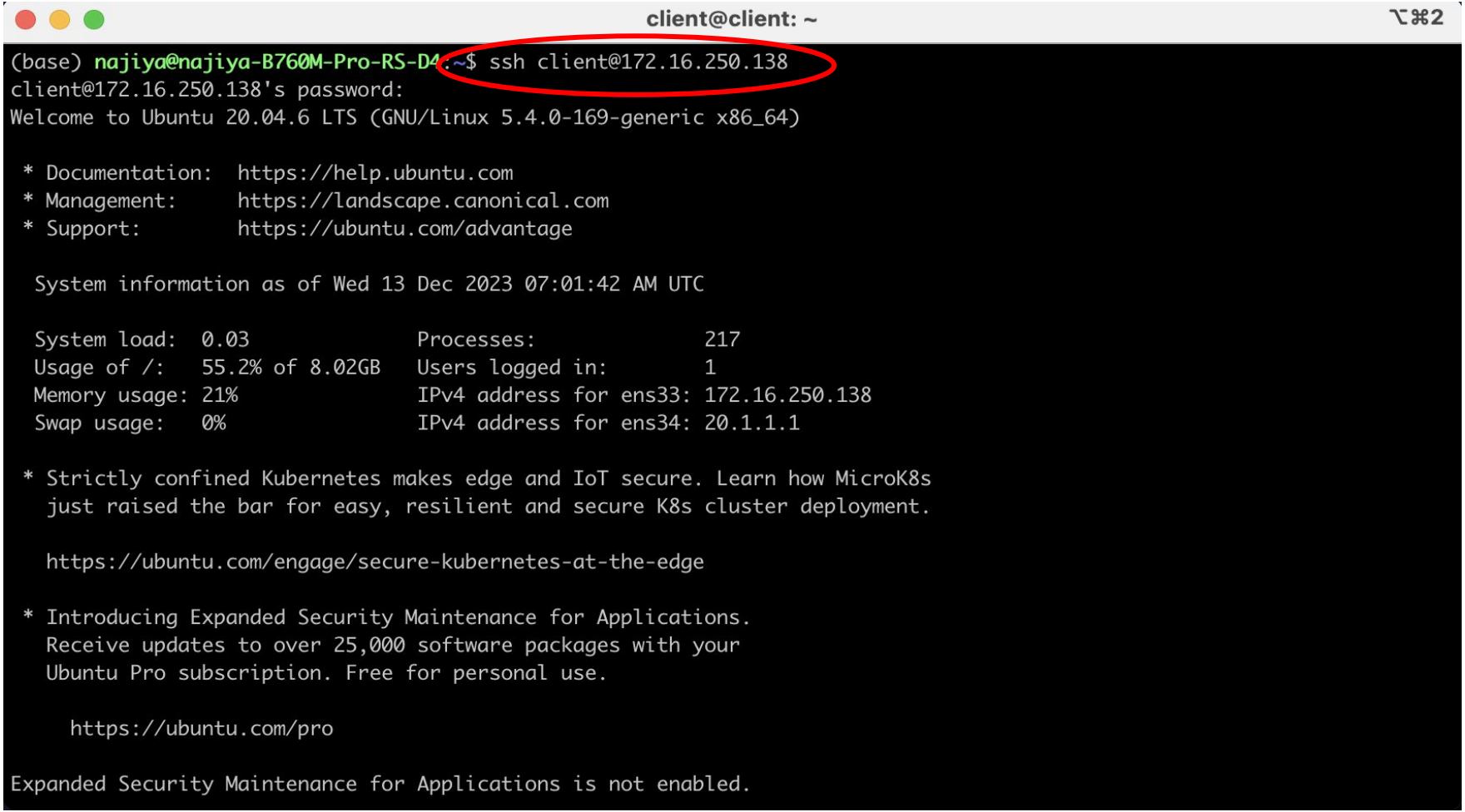

ens34:
40.1.1.1/24
ens34:
20.1.1.1/24
ens34:
20.1.1.2/24
ens36:
40.1.1.2/24
server1
client
gateway
ens34:
40.1.1.3/24
server2


client@client: ~
⌥⌘2
client@client:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.16.250.138 netmask 255.255.255.0 broadcast 172.16.250.255
inet6 fe80::20c:29ff:feb8:2b91 prefixlen 64 scopeid 0x20<link>
ether 00:0c:29:b8:2b:91 txqueuelen 1000 (Ethernet)
RX packets 51241 bytes 44767444 (44.7 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 17654 bytes 2636491 (2.6 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
ens34: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 20.1.1.1 netmask 255.255.255.0 broadcast 20.1.1.255
inet6 fe80::20c:29ff:feb8:2b9b prefixlen 64 scopeid 0x20<link>
ether 00:0c:29:b8:2b:9b txqueuelen 1000 (Ethernet)
RX packets 140517 bytes 9574387 (9.5 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 4655652 bytes 7031785459 (7.0 GB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 528 bytes 110678 (110.6 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 528 bytes 110678 (110.6 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ssh

- Secure remote access and data transfer.
- Syntax: ssh [username]@[hostname or IP address]

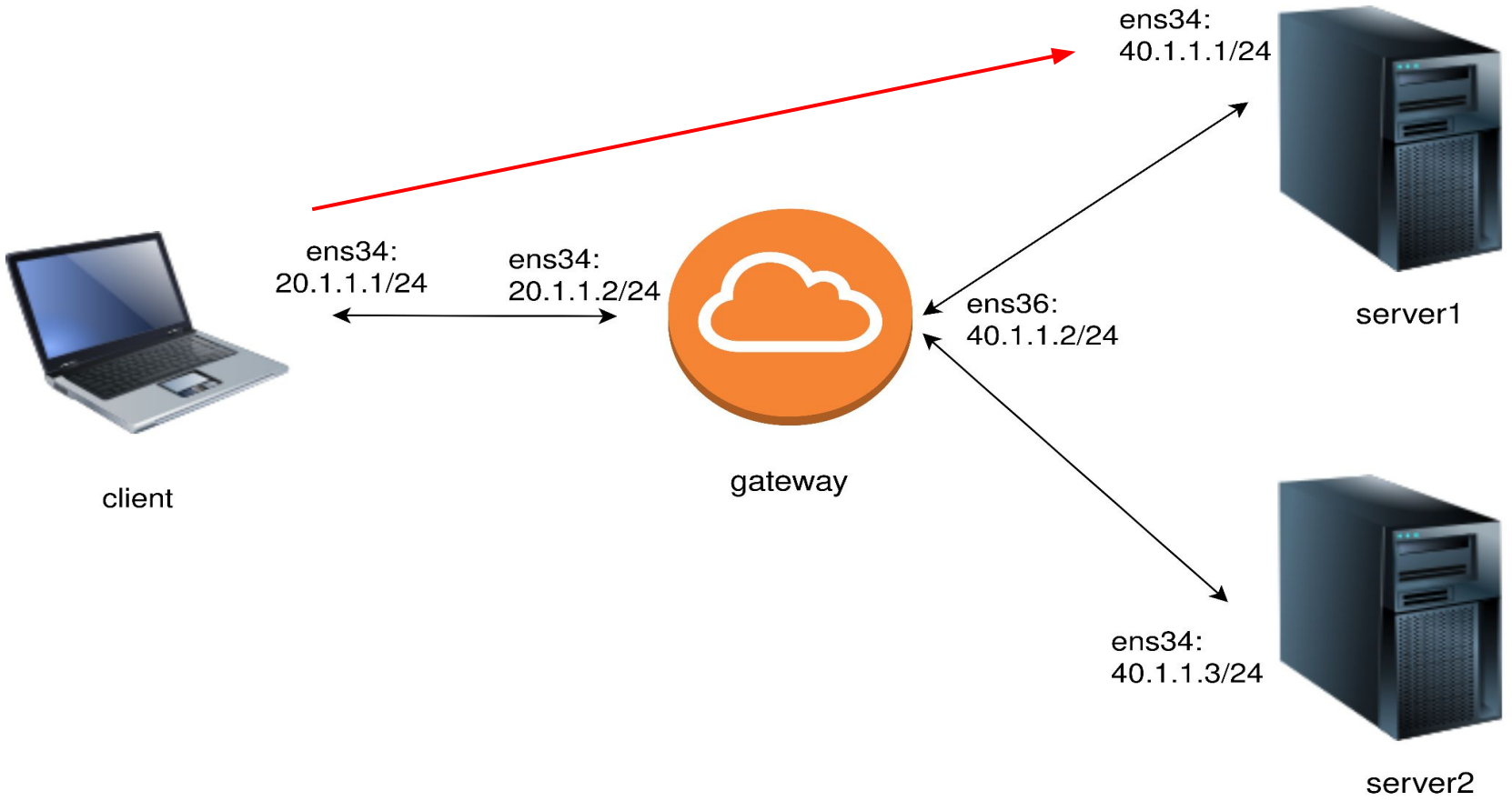
```
sudo apt install openssh-server
```


client@client: ~
⌥⌘2
(base) najiya@najiya-B760M-Pro-RS-D4:~$ ssh client@172.16.250.138
client@172.16.250.138's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-169-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
System information as of Wed 13 Dec 2023 07:01:42 AM UTC
System load: 0.03
Usage of /: 55.2% of 8.02GB
Memory usage: 21%
Swap usage: 0%
Processes: 217
Users logged in: 1
IPv4 address for ens33: 172.16.250.138
IPv4 address for ens34: 20.1.1.1
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
just raised the bar for easy, resilient and secure K8s cluster deployment.
https://ubuntu.com/engage/secure-kubernetes-at-the-edge
* Introducing Expanded Security Maintenance for Applications.
Receive updates to over 25,000 software packages with your
Ubuntu Pro subscription. Free for personal use.
https://ubuntu.com/pro
Expanded Security Maintenance for Applications is not enabled.

ping

- It is network utility tool
- Test the reachability of a host on an Internet Protocol (IP) network.
- Measures the round-trip time from the host to a destination.
- Example: ‘ping www.google.com’ - Sends ICMP packets to the specified website and displays round-trip time and packet loss statistics.

```
ping <ip address>
```

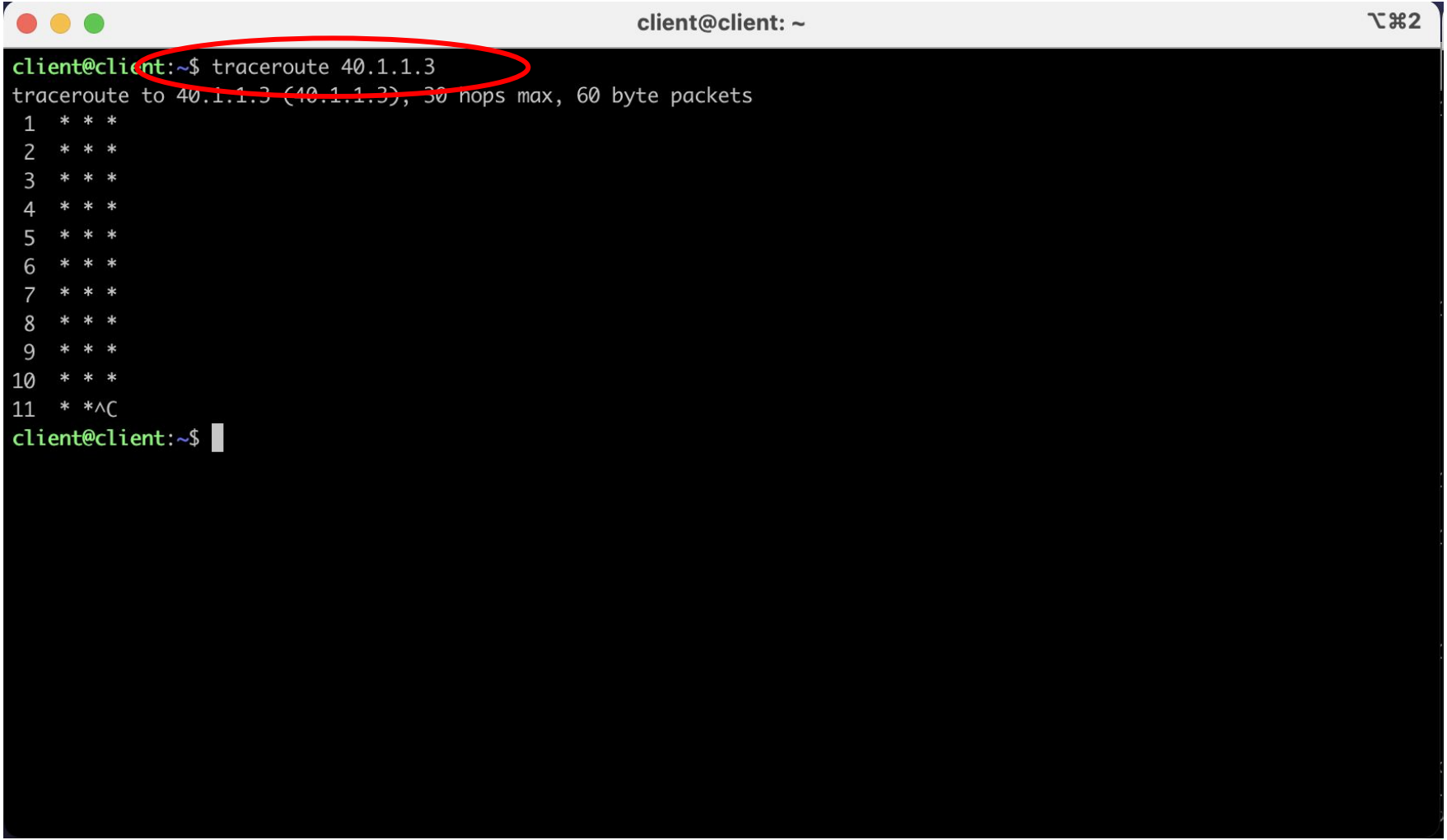

ens34:
40.1.1.1/24
server1
ens34:
20.1.1.1/24
ens34:
20.1.1.2/24
ens36:
40.1.1.2/24
client
gateway
ens34:
40.1.1.3/24
server2


client@client: ~
⌥⌘2
client@client:~$ ping 40.1.1.1
PING 40.1.1.1 (40.1.1.1) 56(84) bytes of data.

traceroute

- Traceroute traces the path of data from one point to another on the Internet.
- Print the route packets trace to network host
- It's a key tool for fixing network issues and grasping network layout.

```
traceroute <ip address>
```

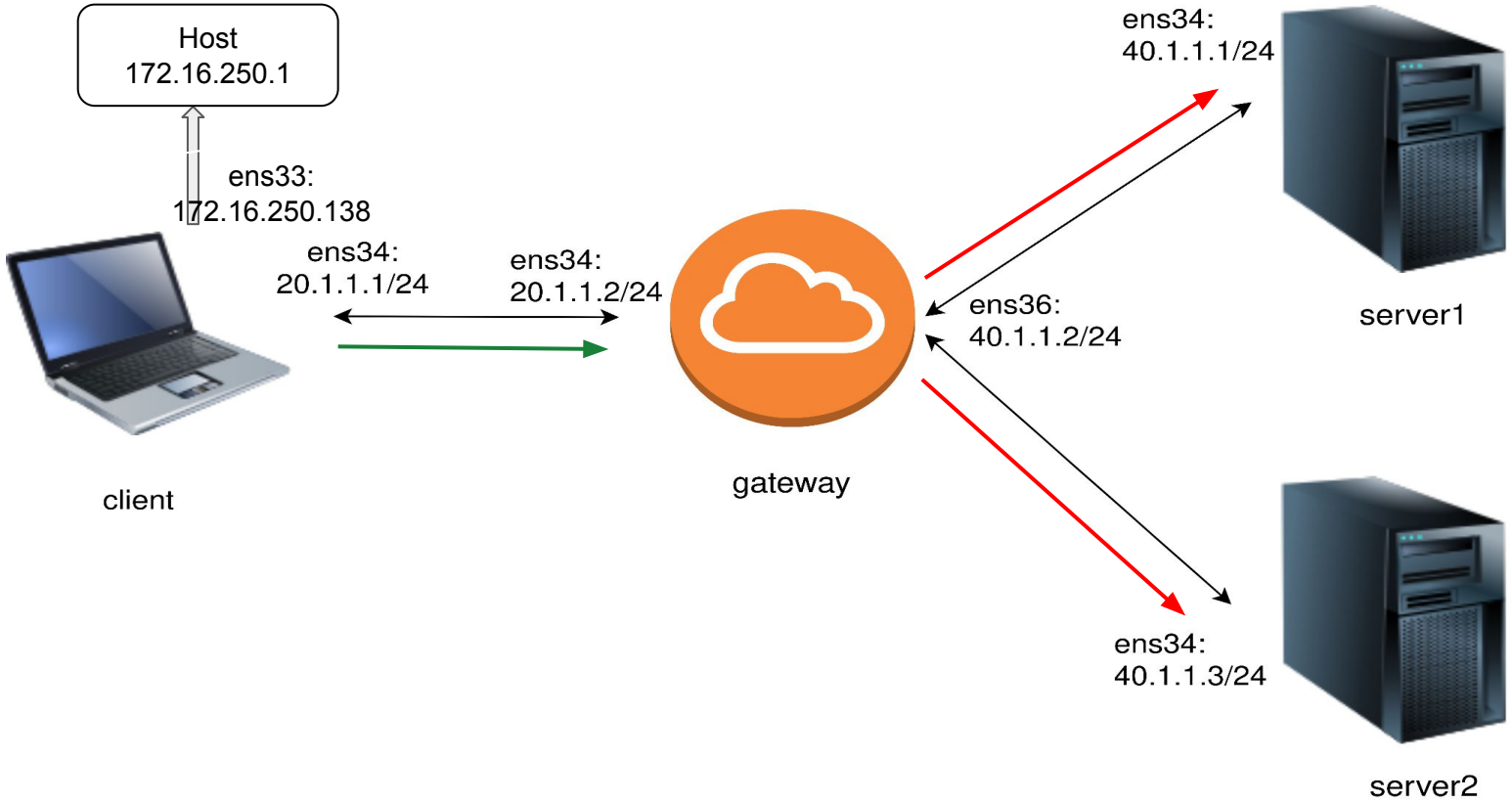

client@client: ~
⌥⌘2
client@client:~$ traceroute 40.1.1.3
traceroute to 40.1.1.3 (40.1.1.3), 30 hops max, 60 byte packets
1 * * *
2 * * *
3 * * *
4 * * *
5 * * *
6 * * *
7 * * *
8 * * *
9 * * *
10 * * *
11 * *^C
client@client:~$

arp (Address Resolution Protocol)

- Networking protocol used to map an IP address to a physical (MAC) address within a local network.
- ARP Request: A broadcast message asking 'Who has this IP address?'"
- ARP Response: The device with the corresponding IP address replies with its MAC address.

```
arp / arp -n
```

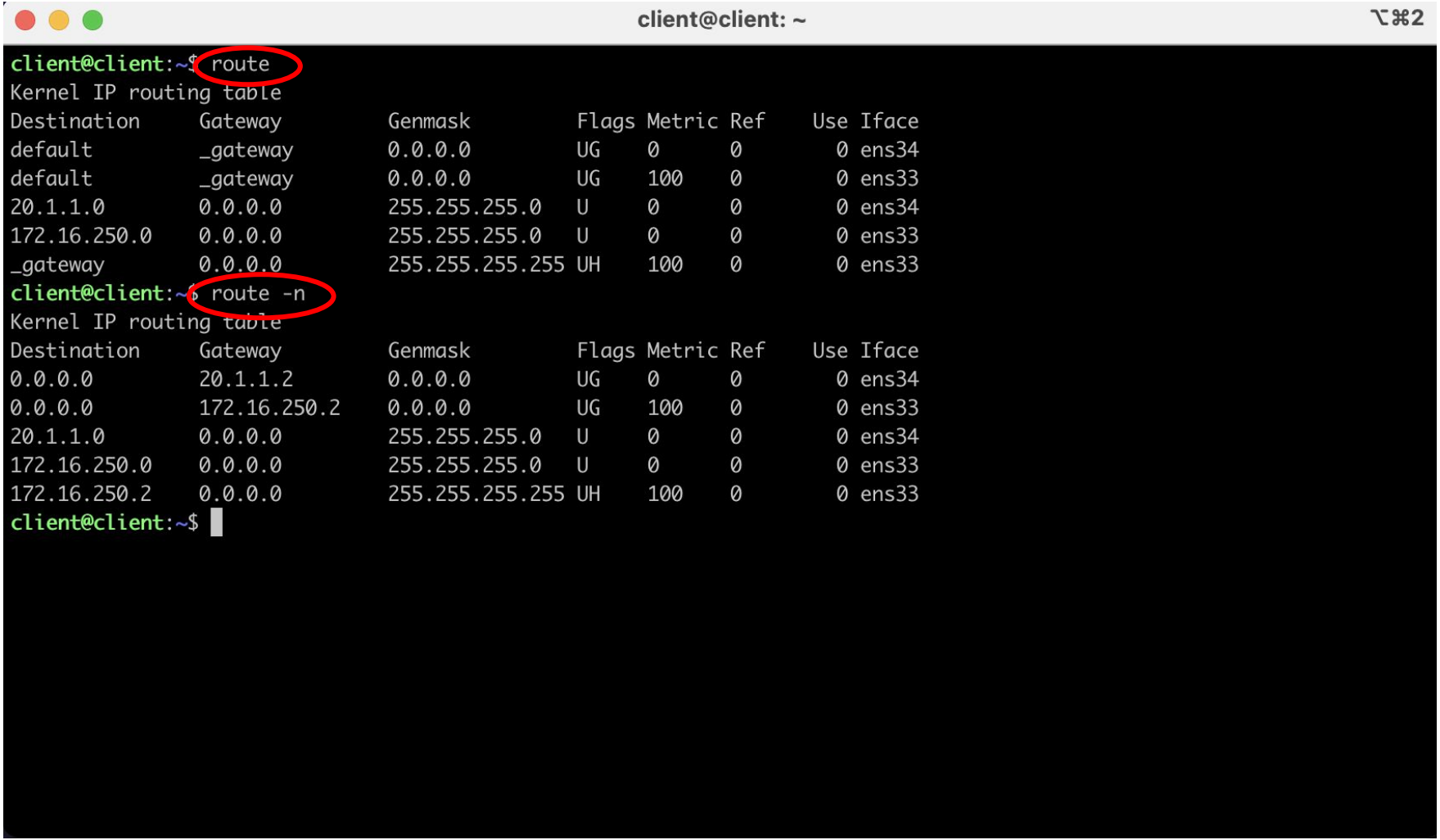

client@client: ~
⌥⌘2
client@client:~$ arp -n
Address HWtype HWaddress Flags Mask Iface
172.16.250.1 ether 00:50:56:c0:00:08 C ens33
20.1.1.2 ether 00:0c:29:ef:6c:8a C ens34
client@client:~$

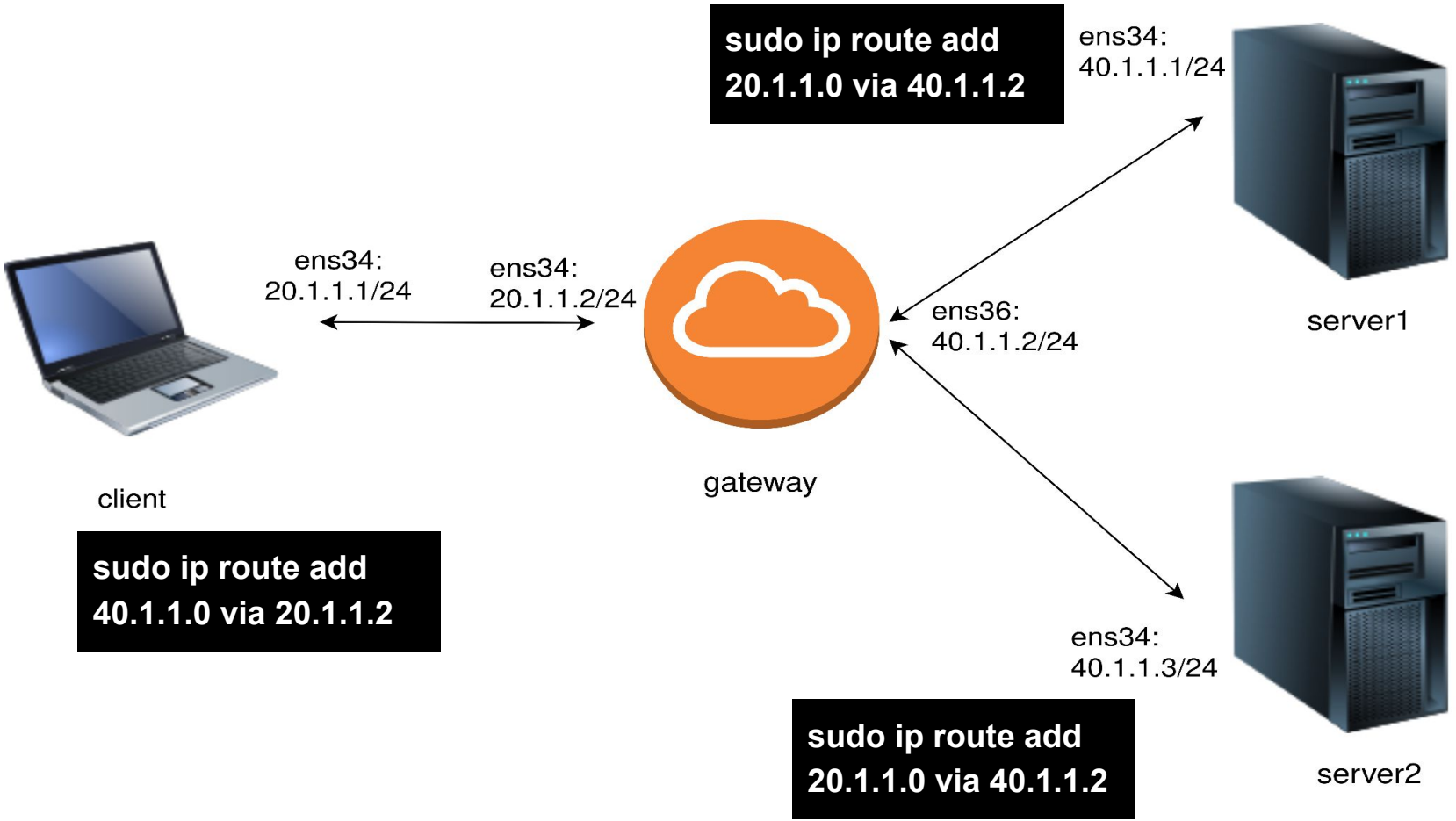

Host
172.16.250.1
ens33:
172.16.250.138
ens34:
20.1.1.1/24
client
ens34:
20.1.1.2/24
gateway
ens34:
40.1.1.1/24
ens36:
40.1.1.2/24
server1
ens34:
40.1.1.3/24
server2

route

- View and manipulate the IP routing table on a computer.
- Allows users to examine the routing table to determine about data packets.
- Addition or removal of static routes to influence network traffic.
- Routing Table is set of rule about network traffic.

```
route / route -n
```


client@client: ~
⌥⌘2
client@client:~$ route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default _gateway 0.0.0.0 UG 0 0 0 ens34
default _gateway 0.0.0.0 UG 100 0 0 ens33
20.1.1.0 0.0.0.0 255.255.255.0 U 0 0 0 ens34
172.16.250.0 0.0.0.0 255.255.255.0 U 0 0 0 ens33
_gateway 0.0.0.0 255.255.255.255 UH 100 0 0 ens33
client@client:~$ route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 20.1.1.2 0.0.0.0 UG 0 0 0 ens34
0.0.0.0 172.16.250.2 0.0.0.0 UG 100 0 0 ens33
20.1.1.0 0.0.0.0 255.255.255.0 U 0 0 0 ens34
172.16.250.0 0.0.0.0 255.255.255.0 U 0 0 0 ens33
172.16.250.2 0.0.0.0 255.255.255.255 UH 100 0 0 ens33
client@client:~$


sudo ip route add
20.1.1.0 via 40.1.1.2
ens34:
40.1.1.1/24
server1
ens34:
20.1.1.1/24
ens34:
20.1.1.2/24
ens36:
40.1.1.2/24
client
gateway
sudo ip route add
40.1.1.0 via 20.1.1.2
ens34:
40.1.1.3/24
server2
sudo ip route add
20.1.1.0 via 40.1.1.2

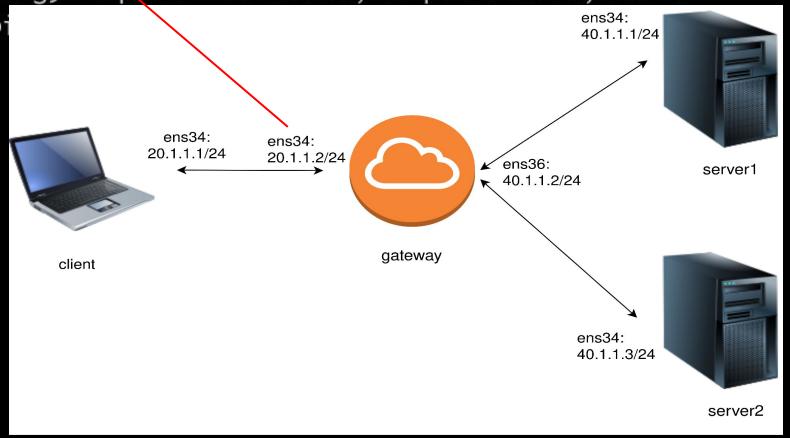
tshark

- Capture and analyze network traffic.
- Provides a versatile tool for network troubleshooting, analysis, and security monitoring.

```
sudo apt install tshark
sudo tshark -i <interface>
```

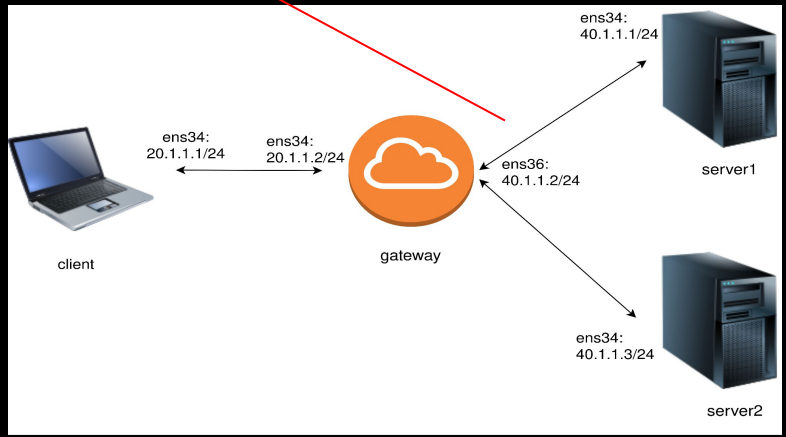
gateway@gateway: ~

```
gateway@gateway:~$ sudo tshark -i ens34
[sudo] password for gateway:
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens34'
    1 0.000000000     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=133/34048, ttl=64
    2 1.023950774     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=134/34304, ttl=64
    3 2.047916786     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=135/34560, ttl=64
    4 3.071999887     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=136/34816, ttl=64
    5 4.095956346     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=137/35072, ttl=64
    6 5.120070482     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=138/35328, ttl=64
    7 6.144020532     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=139/35584, ttl=64
    8 7.168024760     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=140/35840, ttl=64
    9 8.192138789     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (ping) request  id=0x001e, seq=141/36096, ttl=64
   10 9.215998536     20.1.1.1 → 40.1.1.1     ICMP 98 Echo (pi
^C10 packets captured
gateway@gateway:~$
```



gateway@gateway: ~

```
gateway@gateway:~$ sudo tshark -i ens36
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens36'
```

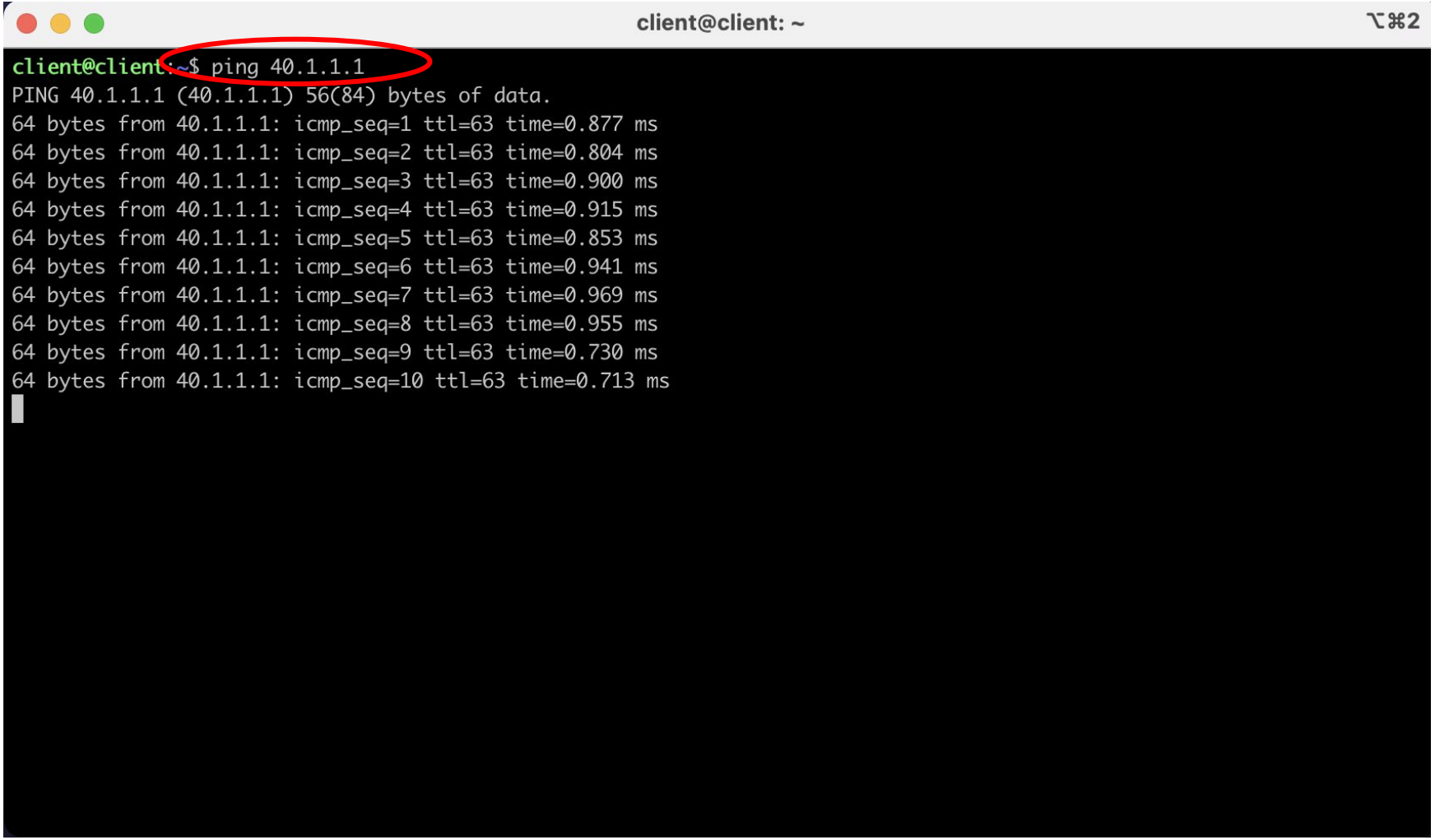


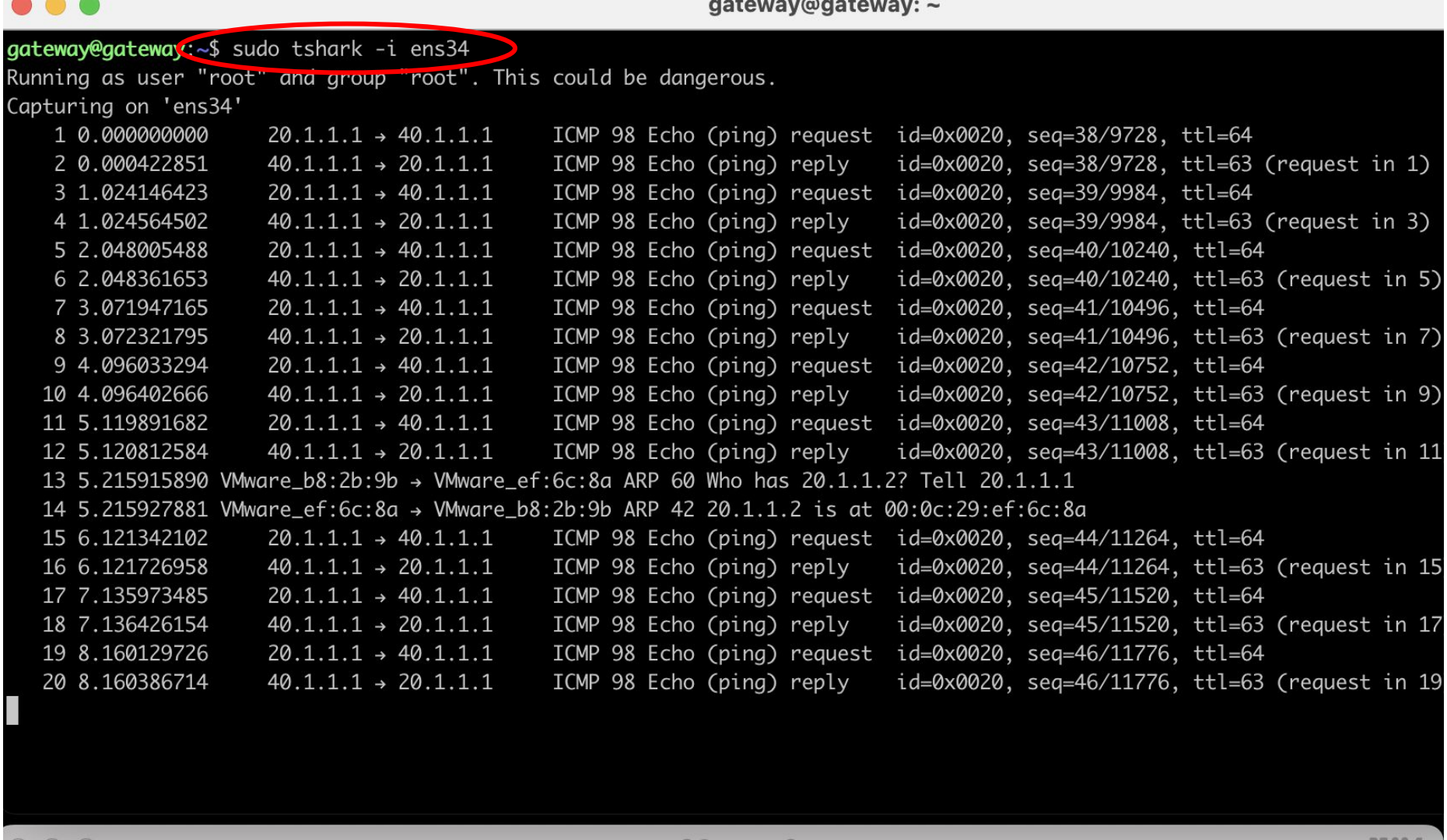
ip forwarding

- Routing of IP packets from one network to another.
- Enables communication between different networks
- Fundamental aspect of data transmission in the Internet.

```
sudo sysctl -w net.ipv4.ip_forward=1
```

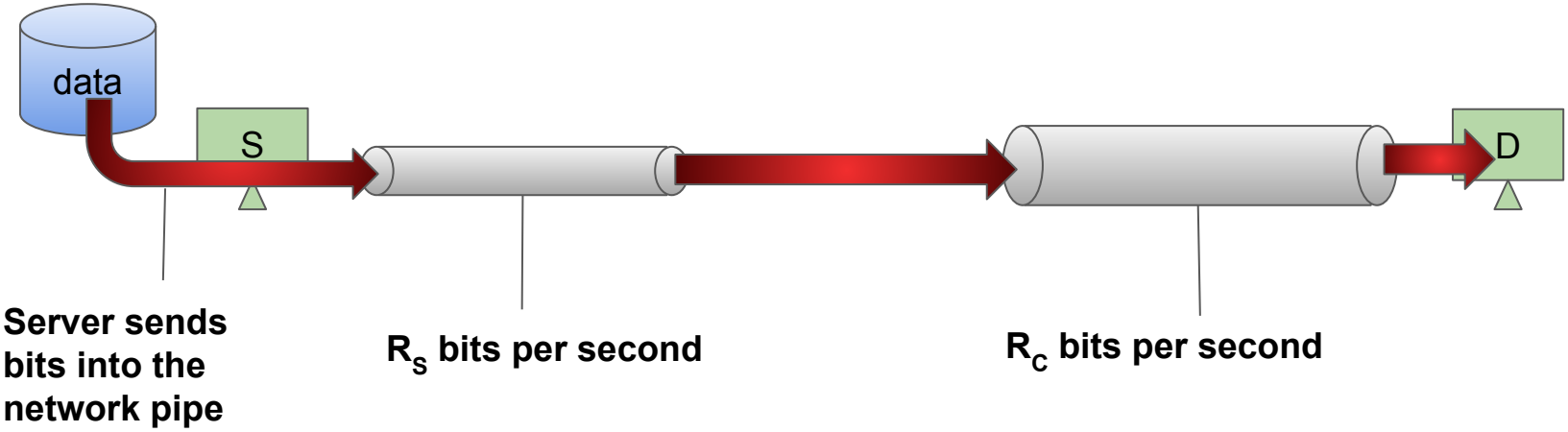

gateway@gateway: ~
gateway@gateway:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
gateway@gateway:~$


client@client: ~
⌥⌘2
client@client:~$ ping 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=0.877 ms
64 bytes from 40.1.1.1: icmp_seq=2 ttl=63 time=0.804 ms
64 bytes from 40.1.1.1: icmp_seq=3 ttl=63 time=0.900 ms
64 bytes from 40.1.1.1: icmp_seq=4 ttl=63 time=0.915 ms
64 bytes from 40.1.1.1: icmp_seq=5 ttl=63 time=0.853 ms
64 bytes from 40.1.1.1: icmp_seq=6 ttl=63 time=0.941 ms
64 bytes from 40.1.1.1: icmp_seq=7 ttl=63 time=0.969 ms
64 bytes from 40.1.1.1: icmp_seq=8 ttl=63 time=0.955 ms
64 bytes from 40.1.1.1: icmp_seq=9 ttl=63 time=0.730 ms
64 bytes from 40.1.1.1: icmp_seq=10 ttl=63 time=0.713 ms


gateway@gateway: ~
gateway@gateway:~$ sudo tshark -i ens34
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens34'
1 0.000000000 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=38/9728, ttl=64
2 0.000422851 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=38/9728, ttl=63 (request in 1)
3 1.024146423 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=39/9984, ttl=64
4 1.024564502 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=39/9984, ttl=63 (request in 3)
5 2.048005488 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=40/10240, ttl=64
6 2.048361653 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=40/10240, ttl=63 (request in 5)
7 3.071947165 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=41/10496, ttl=64
8 3.072321795 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=41/10496, ttl=63 (request in 7)
9 4.096033294 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=42/10752, ttl=64
10 4.096402666 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=42/10752, ttl=63 (request in 9)
11 5.119891682 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=43/11008, ttl=64
12 5.120812584 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=43/11008, ttl=63 (request in 11
13 5.215915890 VMware_b8:2b:9b → VMware_ef:6c:8a ARP 60 Who has 20.1.1.2? Tell 20.1.1.1
14 5.215927881 VMware_ef:6c:8a → VMware_b8:2b:9b ARP 42 20.1.1.2 is at 00:0c:29:ef:6c:8a
15 6.121342102 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=44/11264, ttl=64
16 6.121726958 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=44/11264, ttl=63 (request in 15
17 7.135973485 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=45/11520, ttl=64
18 7.136426154 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=45/11520, ttl=63 (request in 17
19 8.160129726 20.1.1.1 → 40.1.1.1 ICMP 98 Echo (ping) request id=0x0020, seq=46/11776, ttl=64
20 8.160386714 40.1.1.1 → 20.1.1.1 ICMP 98 Echo (ping) reply id=0x0020, seq=46/11776, ttl=63 (request in 19

Measurement of throughput between client and server

Throughput



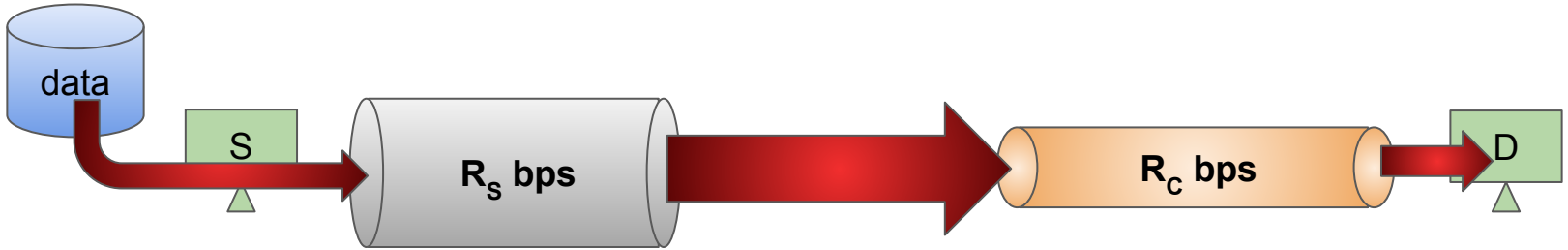
Throughput:

Rate at which bits are transferred between the sender and receiver

Throughput (contd.)

$R_S < R_C$ What is the average end-end throughput?

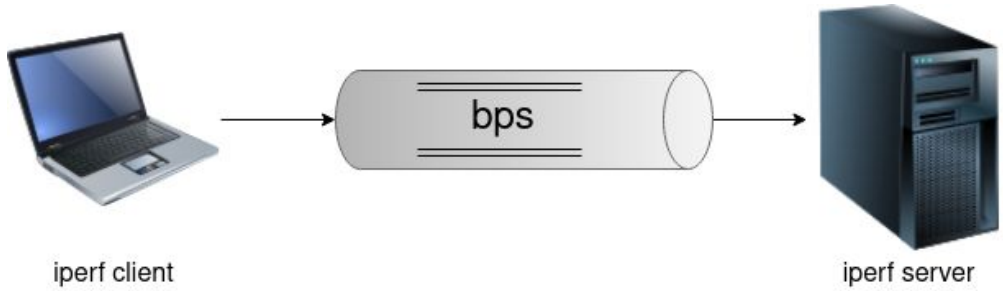
$R_S > R_C$ What is the average end-end throughput?



Bottleneck link on the path constraints end-end throughput

iperf

- Measures network performance by generating TCP and UDP data streams between two systems.
- Evaluates the bandwidth and quality of a network connection.
- Troubleshooting, optimizing, and assessing network performance.

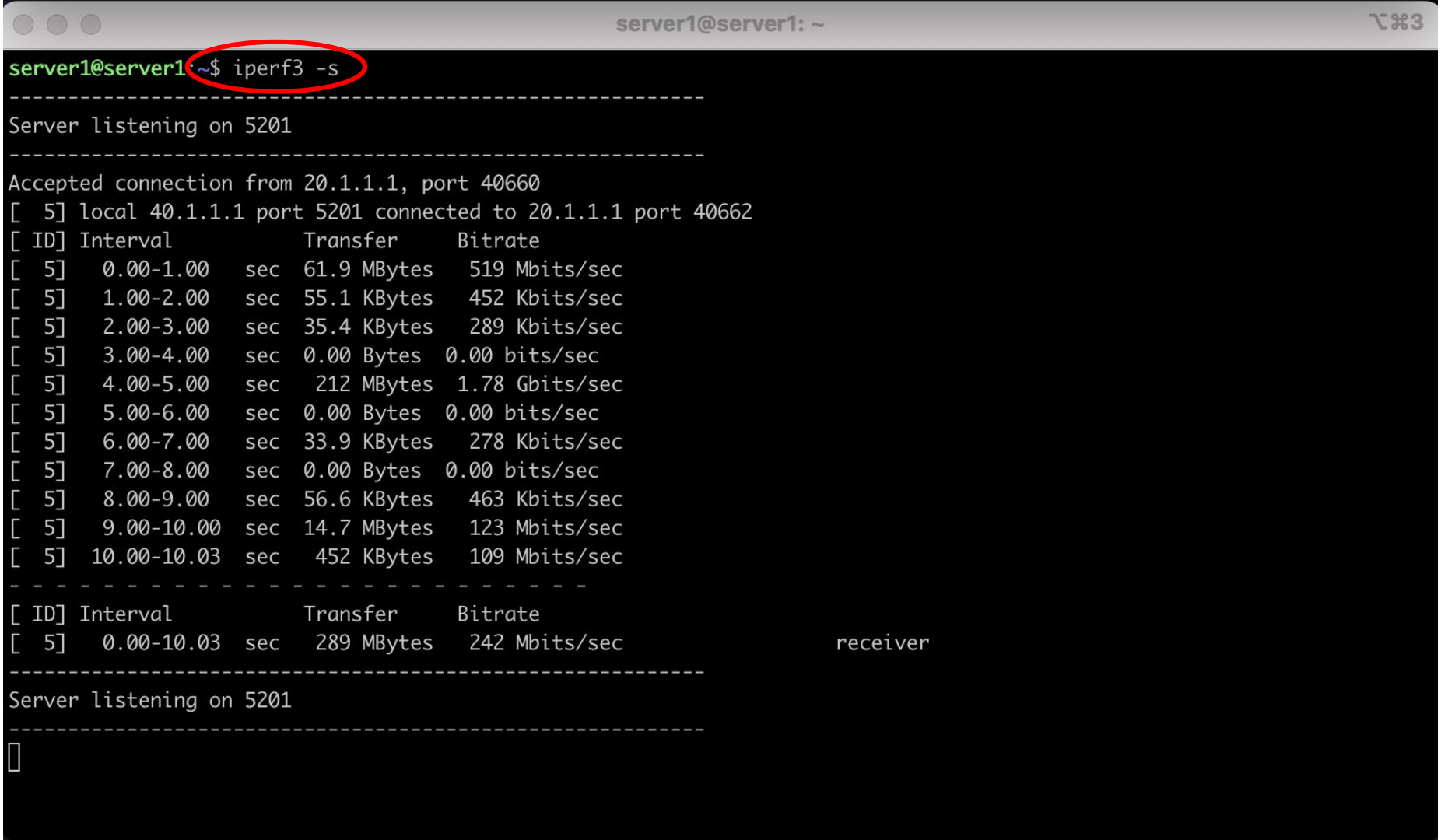


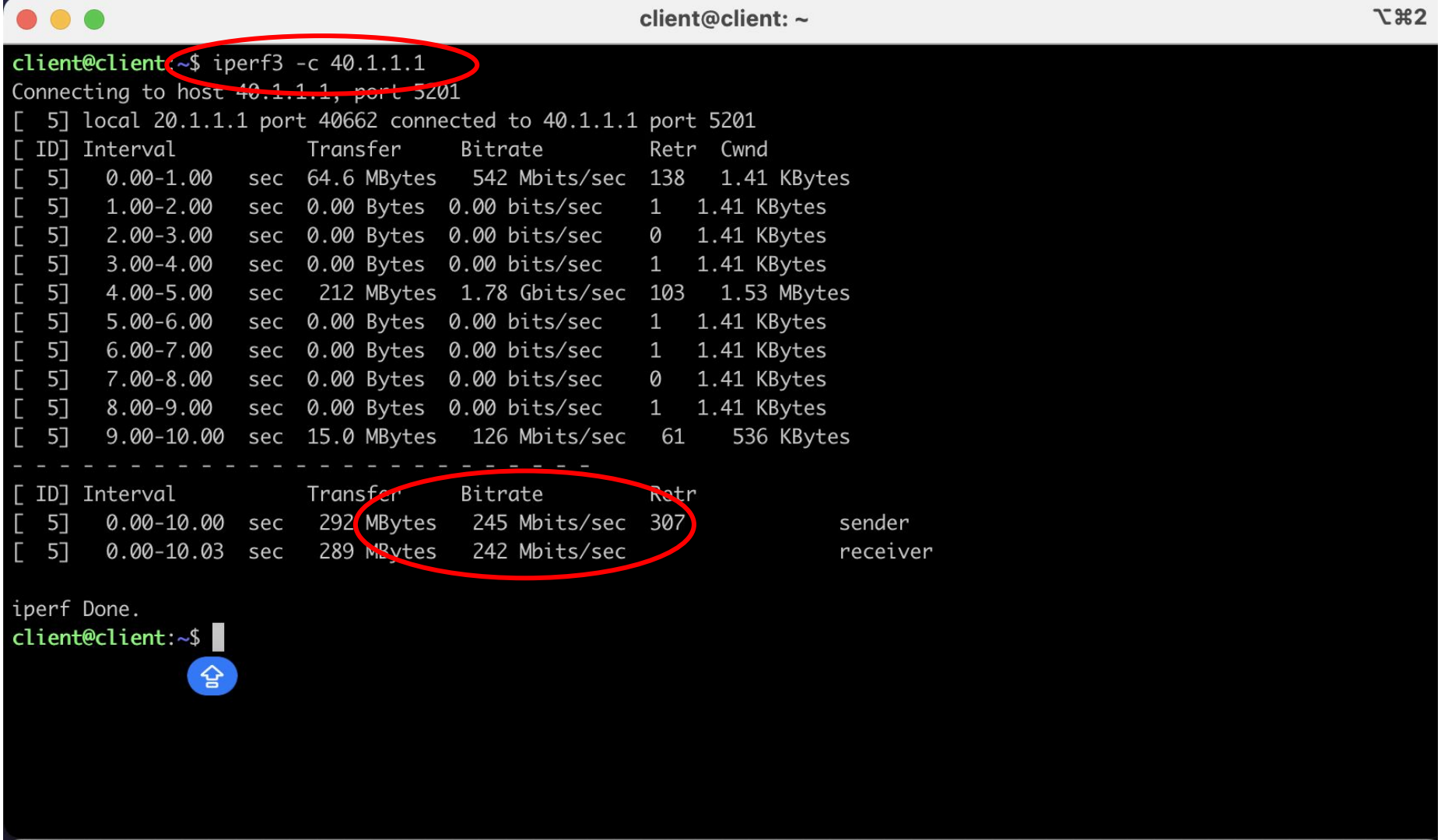
iperf

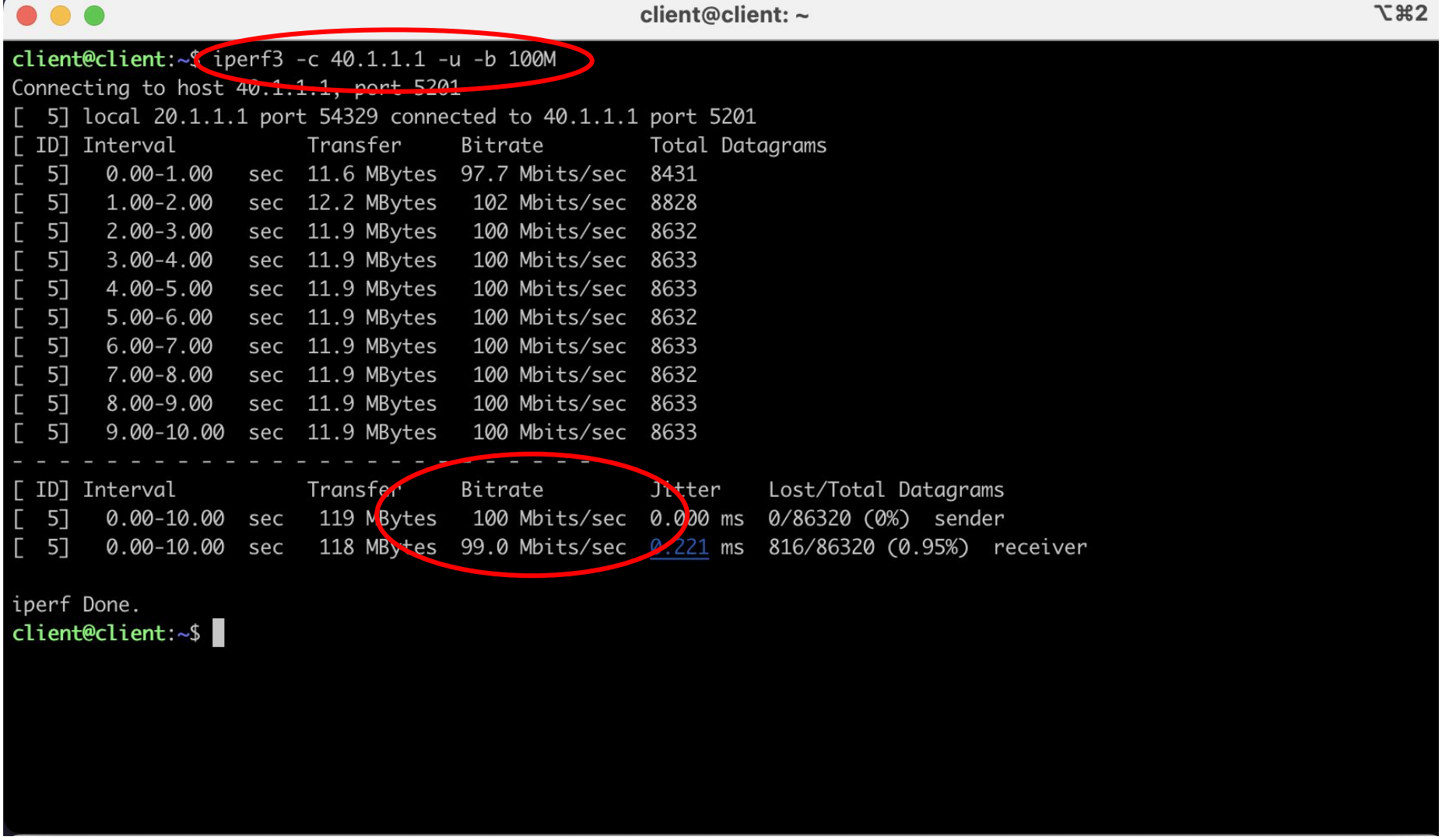
- Measures network performance by generating TCP and UDP data streams between two systems.
- Evaluates the bandwidth and quality of a network connection.
- Troubleshooting, optimizing, and assessing network performance.

```
sudo apt update
sudo apt-get install iperf3
```

```
iperf3 -s
iperf3 -c 40.1.1.1
iperf3 -c 40.1.1.1 -u
```


server1@server1: ~
server1@server1:~$ iperf3 -s
Server listening on 5201
Accepted connection from 20.1.1.1, port 40660
[5] local 40.1.1.1 port 5201 connected to 20.1.1.1 port 40662
[ID] Interval Transfer Bitrate
[5] 0.00-1.00 sec 61.9 MBytes 519 Mbits/sec
[5] 1.00-2.00 sec 55.1 KBytes 452 Kbits/sec
[5] 2.00-3.00 sec 35.4 KBytes 289 Kbits/sec
[5] 3.00-4.00 sec 0.00 Bytes 0.00 bits/sec
[5] 4.00-5.00 sec 212 MBytes 1.78 Gbits/sec
[5] 5.00-6.00 sec 0.00 Bytes 0.00 bits/sec
[5] 6.00-7.00 sec 33.9 KBytes 278 Kbits/sec
[5] 7.00-8.00 sec 0.00 Bytes 0.00 bits/sec
[5] 8.00-9.00 sec 56.6 KBytes 463 Kbits/sec
[5] 9.00-10.00 sec 14.7 MBytes 123 Mbits/sec
[5] 10.00-10.03 sec 452 KBytes 109 Mbits/sec
[ID] Interval Transfer Bitrate
[5] 0.00-10.03 sec 289 MBytes 242 Mbits/sec receiver
Server listening on 5201


client@client: ~
⌥⌘2
client@client:~$ iperf3 -c 40.1.1.1
Connecting to host 40.1.1.1, port 5201
[5] local 20.1.1.1 port 40662 connected to 40.1.1.1 port 5201
[ID] Interval Transfer Bitrate Retr Cwnd
[5] 0.00-1.00 sec 64.6 MBytes 542 Mbits/sec 138 1.41 KBytes
[5] 1.00-2.00 sec 0.00 Bytes 0.00 bits/sec 1 1.41 KBytes
[5] 2.00-3.00 sec 0.00 Bytes 0.00 bits/sec 0 1.41 KBytes
[5] 3.00-4.00 sec 0.00 Bytes 0.00 bits/sec 1 1.41 KBytes
[5] 4.00-5.00 sec 212 MBytes 1.78 Gbits/sec 103 1.53 MBytes
[5] 5.00-6.00 sec 0.00 Bytes 0.00 bits/sec 1 1.41 KBytes
[5] 6.00-7.00 sec 0.00 Bytes 0.00 bits/sec 1 1.41 KBytes
[5] 7.00-8.00 sec 0.00 Bytes 0.00 bits/sec 0 1.41 KBytes
[5] 8.00-9.00 sec 0.00 Bytes 0.00 bits/sec 1 1.41 KBytes
[5] 9.00-10.00 sec 15.0 MBytes 126 Mbits/sec 61 536 KBytes
- -
[ID] Interval Transfer Bitrate Retr
[5] 0.00-10.00 sec 292 MBytes 245 Mbits/sec 307 sender
[5] 0.00-10.03 sec 289 MBytes 242 Mbits/sec receiver
iperf Done.
client@client:~$


client@client: ~
⌥⌘2
client@client:~$ iperf3 -c 40.1.1.1 -u -b 100M
Connecting to host 40.1.1.1, port 5201
[5] local 20.1.1.1 port 54329 connected to 40.1.1.1 port 5201
[ID] Interval Transfer Bitrate Total Datagrams
[5] 0.00-1.00 sec 11.6 MBytes 97.7 Mbits/sec 8431
[5] 1.00-2.00 sec 12.2 MBytes 102 Mbits/sec 8828
[5] 2.00-3.00 sec 11.9 MBytes 100 Mbits/sec 8632
[5] 3.00-4.00 sec 11.9 MBytes 100 Mbits/sec 8633
[5] 4.00-5.00 sec 11.9 MBytes 100 Mbits/sec 8633
[5] 5.00-6.00 sec 11.9 MBytes 100 Mbits/sec 8632
[5] 6.00-7.00 sec 11.9 MBytes 100 Mbits/sec 8633
[5] 7.00-8.00 sec 11.9 MBytes 100 Mbits/sec 8632
[5] 8.00-9.00 sec 11.9 MBytes 100 Mbits/sec 8633
[5] 9.00-10.00 sec 11.9 MBytes 100 Mbits/sec 8633
[ID] Interval Transfer Bitrate Jitter Lost/Total Datagrams
[5] 0.00-10.00 sec 119 MBytes 100 Mbits/sec 0.000 ms 0/86320 (0%) sender
[5] 0.00-10.00 sec 118 MBytes 99.0 Mbits/sec 0.221 ms 816/86320 (0.95%) receiver
iperf Done.
client@client:~$

netstat

- Display network connections, routing tables, interface statistics, masquerade connections, and multicast memberships on Unix-like operating systems.
- Provides a snapshot of current network activity and configuration.
- Useful for diagnosing network-related issues and monitoring network connections.

```
netstat
netstat -a
netstat -I
```

```
gateway@gateway:~$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 gateway:ssh             najiya-B760M-Pro-:53176 ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    40460    /run/user/1000/systemd/notify
unix  3      [ ]         DGRAM                    28754    /run/systemd/notify
unix  2      [ ]         DGRAM                    28771    /run/systemd/journal/syslog
unix  8      [ ]         DGRAM                    28781    /run/systemd/journal/dev-log
unix  9      [ ]         DGRAM                    28785    /run/systemd/journal/socket
unix  3      [ ]         STREAM     CONNECTED     38209
unix  3      [ ]         STREAM     CONNECTED     34837    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     32651
unix  3      [ ]         STREAM     CONNECTED     37991    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     35286
unix  3      [ ]         STREAM     CONNECTED     37987    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     35806
unix  3      [ ]         DGRAM                    34295
unix  3      [ ]         STREAM     CONNECTED     37719
unix  3      [ ]         STREAM     CONNECTED     40465    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     41240
unix  3      [ ]         STREAM     CONNECTED     38286    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     38413    /run/dbus/system_bus_socket
unix  2      [ ]         DGRAM                    38305
unix  3      [ ]         STREAM     CONNECTED     35815    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     36823
```

server2@server2: ~

```
gateway@gateway:~$ netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 gateway:ssh             najiya-B760M-Pro-:53176 ESTABLISHED
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
udp        0      0 localhost:domain        0.0.0.0:*
udp        0      0 gateway:bootpc          0.0.0.0:*
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     30253    /run/systemd/journal/io.systemd.journal
unix  2      [ ACC ]     SEQPACKET  LISTENING     28788    /run/udev/control
unix  2      [ ]         DGRAM                    40460    /run/user/1000/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     40463    /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     40470    /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     40471    /run/user/1000/gnupg/S.dirmngr
unix  2      [ ACC ]     STREAM     LISTENING     40472    /run/user/1000/gnupg/S.gpg-agent.browser
unix  2      [ ACC ]     STREAM     LISTENING     35288    /var/snap/lxd/common/lxd/unix.socket
unix  2      [ ACC ]     STREAM     LISTENING     40473    /run/user/1000/gnupg/S.gpg-agent.extra
unix  2      [ ACC ]     STREAM     LISTENING     28770    @/org/kernel/linux/storage/multipathd
unix  2      [ ACC ]     STREAM     LISTENING     40474    /run/user/1000/gnupg/S.gpg-agent.ssh
```

Demo Session 2

Rate limiting by the gateway

- tc command

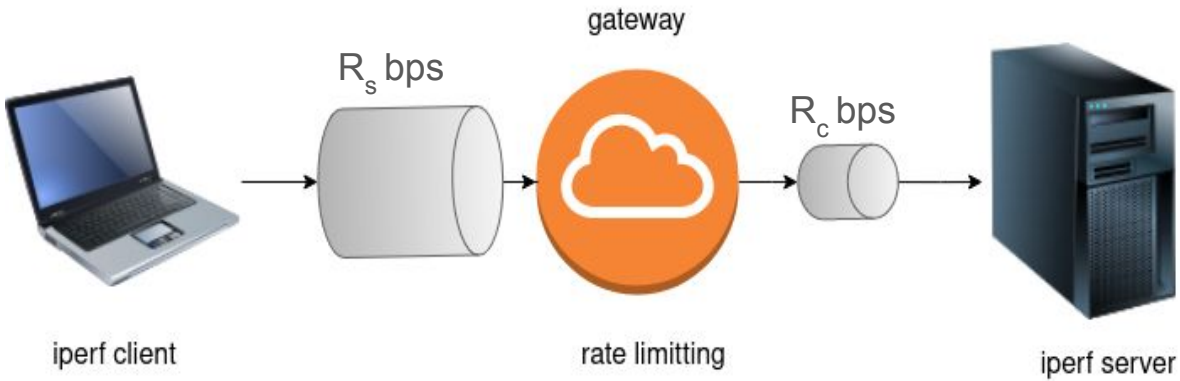
Iptables

- Filter
- NAT
  - SNAT
  - DNAT
- Load Balancing

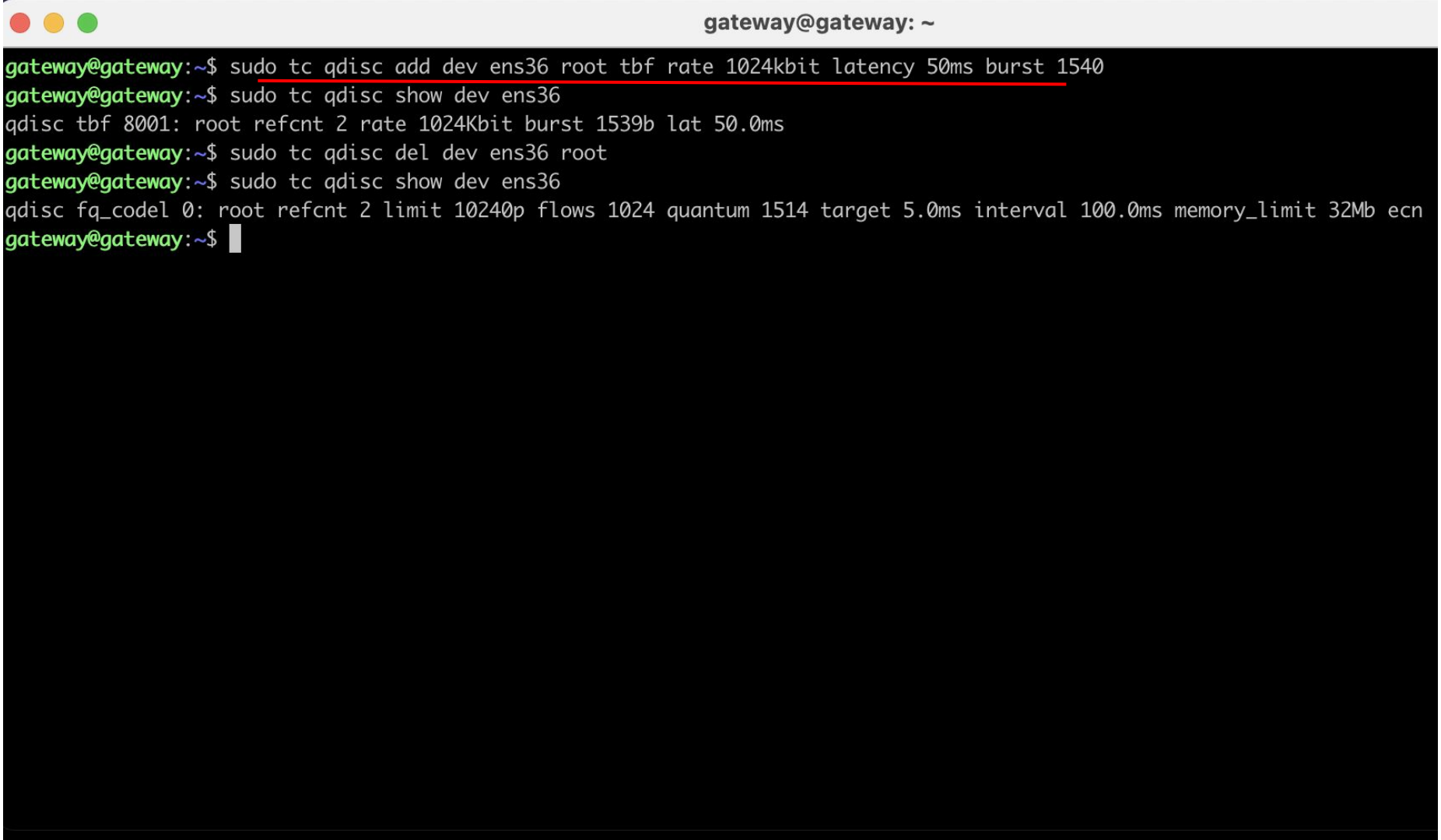
Rate limiting by the gateway

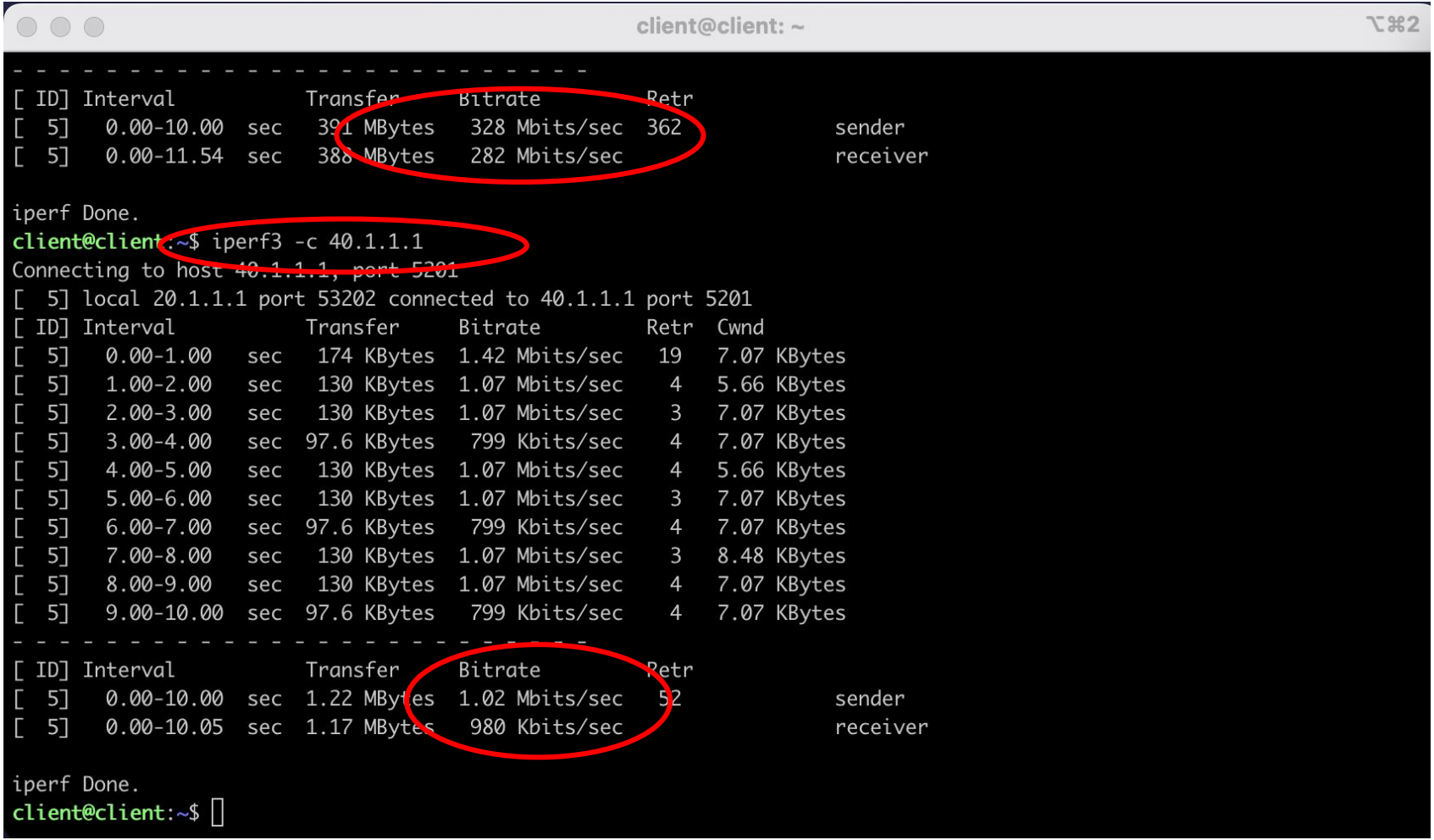
tc (Traffic control)

- Helps control network traffic by managing bandwidth, latency, and packet loss.
- tc qdisc for configuring queuing disciplines.
- “tc” to limit bandwidth: tc qdisc add dev eth0 root tbf rate 1mbit burst 10kb latency 70ms.



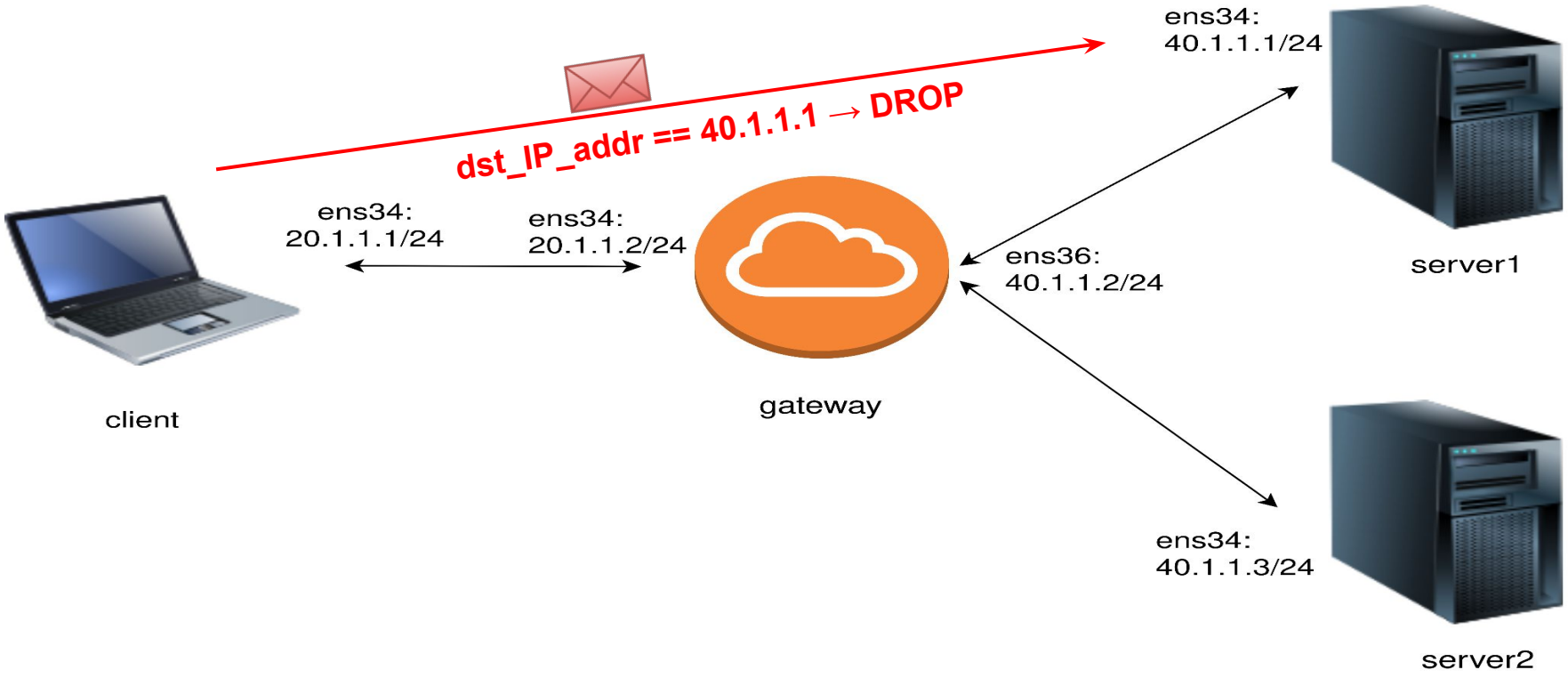
```
sudo tc qdisc add dev ens36 root tbf rate 1024kbit latency 50ms burst 1540

sudo tc qdisc show dev ens36

sudo tc qdisc del dev ens36 root
```


gateway@gateway: ~
gateway@gateway:~$ sudo tc qdisc add dev ens36 root tbf rate 1024kbit latency 50ms burst 1540
gateway@gateway:~$ sudo tc qdisc show dev ens36
qdisc tbf 8001: root refcnt 2 rate 1024Kbit burst 1539b lat 50.0ms
gateway@gateway:~$ sudo tc qdisc del dev ens36 root
gateway@gateway:~$ sudo tc qdisc show dev ens36
qdisc fq_codel 0: root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms interval 100.0ms memory_limit 32Mb ecn
gateway@gateway:~$

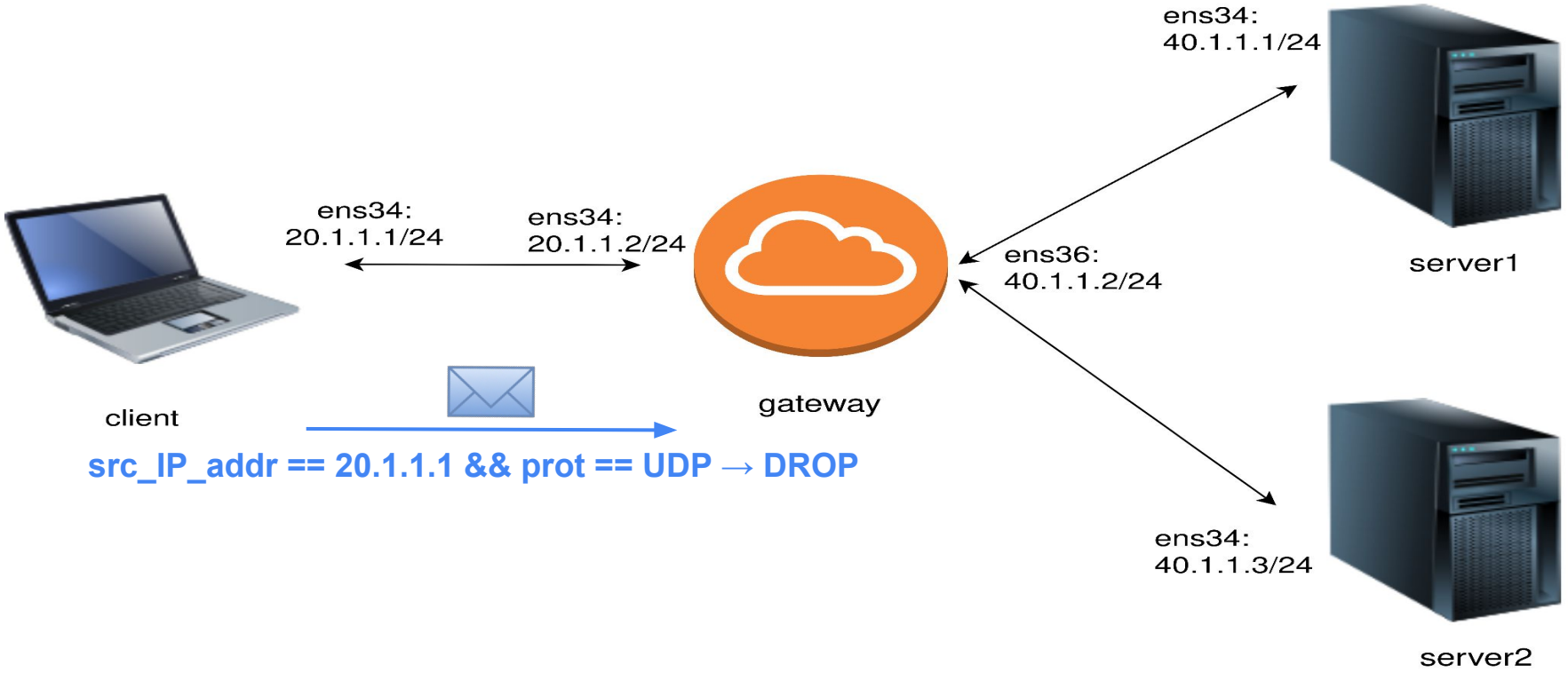

client@client: ~
⌥⌘2
- -
[ID] Interval Transfer Bitrate Retr
[5] 0.00-10.00 sec 391 MBytes 328 Mbits/sec 362 sender
[5] 0.00-11.54 sec 388 MBytes 282 Mbits/sec receiver
iperf Done.
client@client:~$ iperf3 -c 40.1.1.1
Connecting to host 40.1.1.1, port 5201
[5] local 20.1.1.1 port 53202 connected to 40.1.1.1 port 5201
[ID] Interval Transfer Bitrate Retr Cwnd
[5] 0.00-1.00 sec 174 KBytes 1.42 Mbits/sec 19 7.07 KBytes
[5] 1.00-2.00 sec 130 KBytes 1.07 Mbits/sec 4 5.66 KBytes
[5] 2.00-3.00 sec 130 KBytes 1.07 Mbits/sec 3 7.07 KBytes
[5] 3.00-4.00 sec 97.6 KBytes 799 Kbits/sec 4 7.07 KBytes
[5] 4.00-5.00 sec 130 KBytes 1.07 Mbits/sec 4 5.66 KBytes
[5] 5.00-6.00 sec 130 KBytes 1.07 Mbits/sec 3 7.07 KBytes
[5] 6.00-7.00 sec 97.6 KBytes 799 Kbits/sec 4 7.07 KBytes
[5] 7.00-8.00 sec 130 KBytes 1.07 Mbits/sec 3 8.48 KBytes
[5] 8.00-9.00 sec 130 KBytes 1.07 Mbits/sec 4 7.07 KBytes
[5] 9.00-10.00 sec 97.6 KBytes 799 Kbits/sec 4 7.07 KBytes
- -
[ID] Interval Transfer Bitrate Retr
[5] 0.00-10.00 sec 1.22 MBytes 1.02 Mbits/sec 52 sender
[5] 0.00-10.05 sec 1.17 MBytes 980 Kbits/sec receiver
iperf Done.
client@client:~$

Netfilter

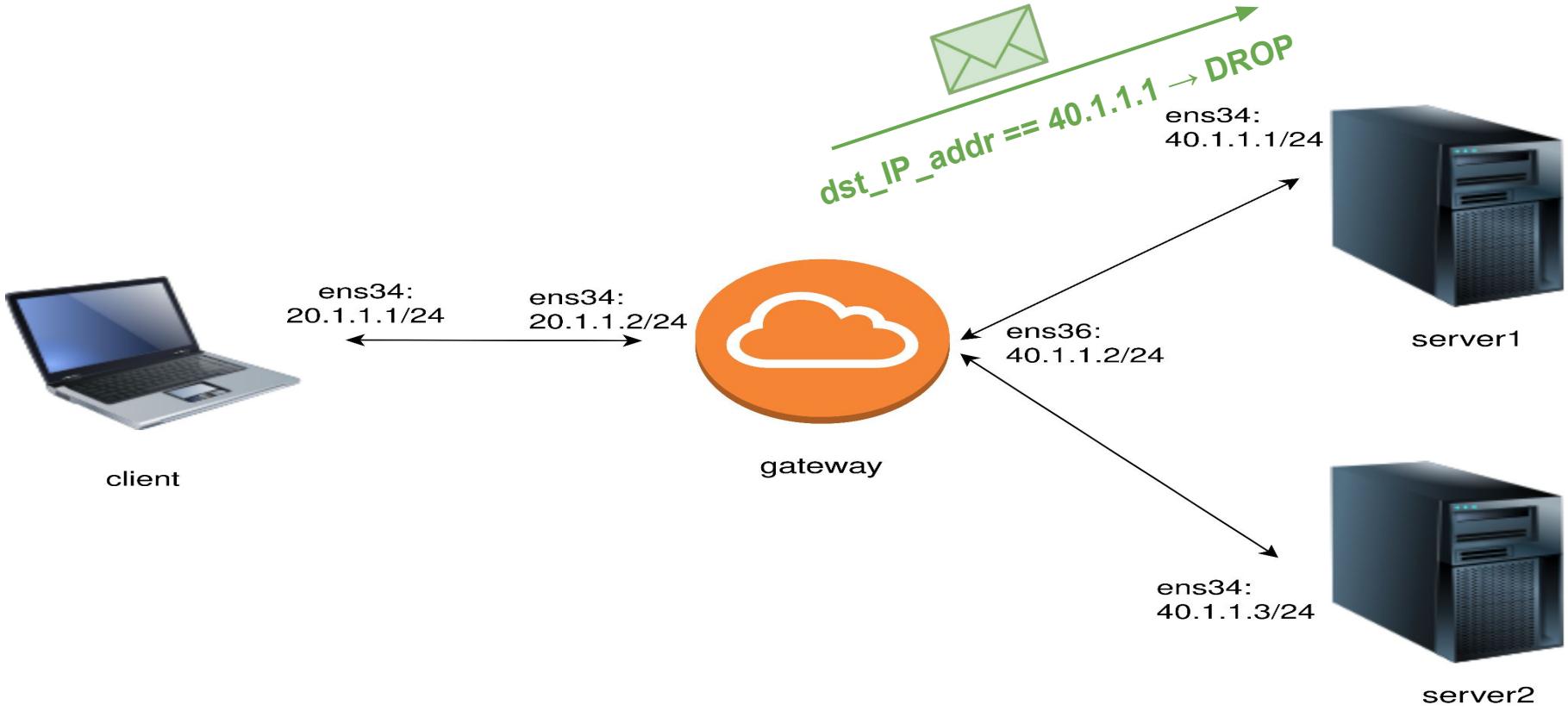
Objective 1 for traffic filtering using netfilter:



Objective 2 for traffic filtering using netfilter:



Objective 3 for traffic filtering using netfilter:

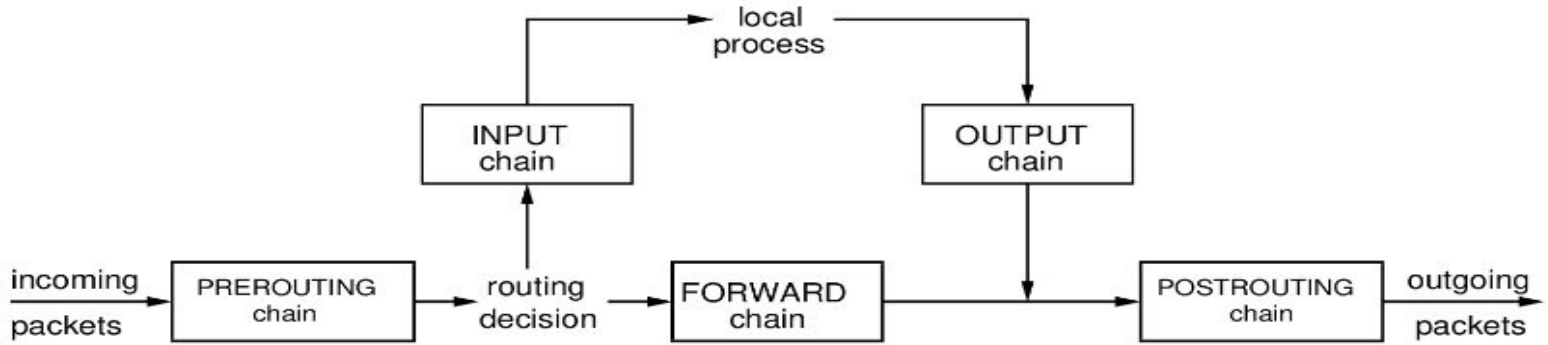


Netfilter

- Known as Linux kernel packet filter framework
- Contains **hooks (chains)** to configure rules at different locations of linux network stack.
- Uses **iptables** software to become a router capable of Firewall, NAT or load balancer.

Netfilter Hooks/Chains:

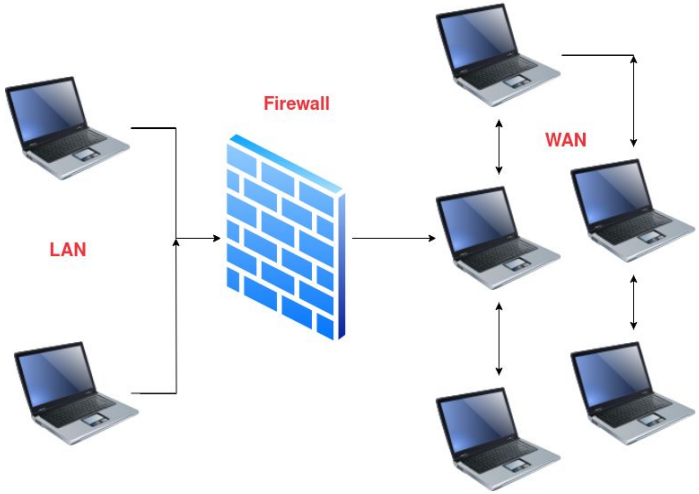
- **Hooks/Chains :** Collection of rules applied to incoming or outgoing packets
  - **Prerouting Chain :** Applies to packets arriving on network.
  - **Input Chain :** Applies to packets right before sending them to local process
  - **Output Chain :** Applies to packets generated by local process.
  - **Postrouting Chain :** Applies to packets leaving the network interface
  - **Forward Chain :** Applies to all packets routed via this host

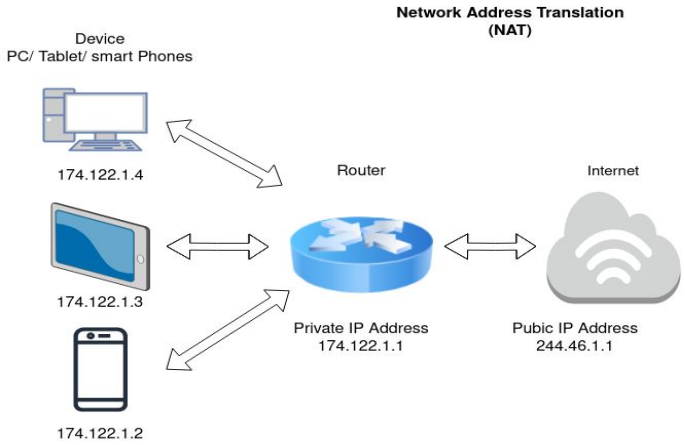


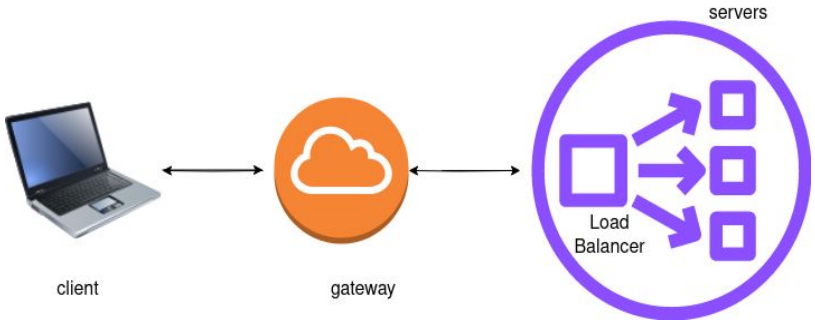
Netfilter Hooks Block Diagram

What we can configure with iptables

- Firewall / packet filter
- NAT
- Load balancing







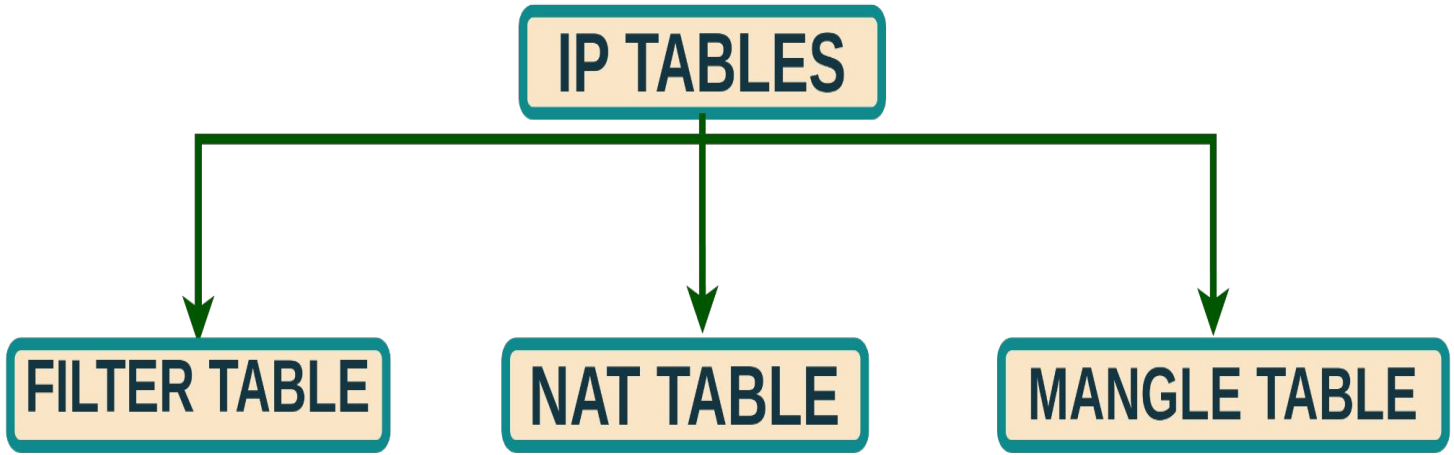
iptables

- User space command line program to configure Linux Kernel firewall.

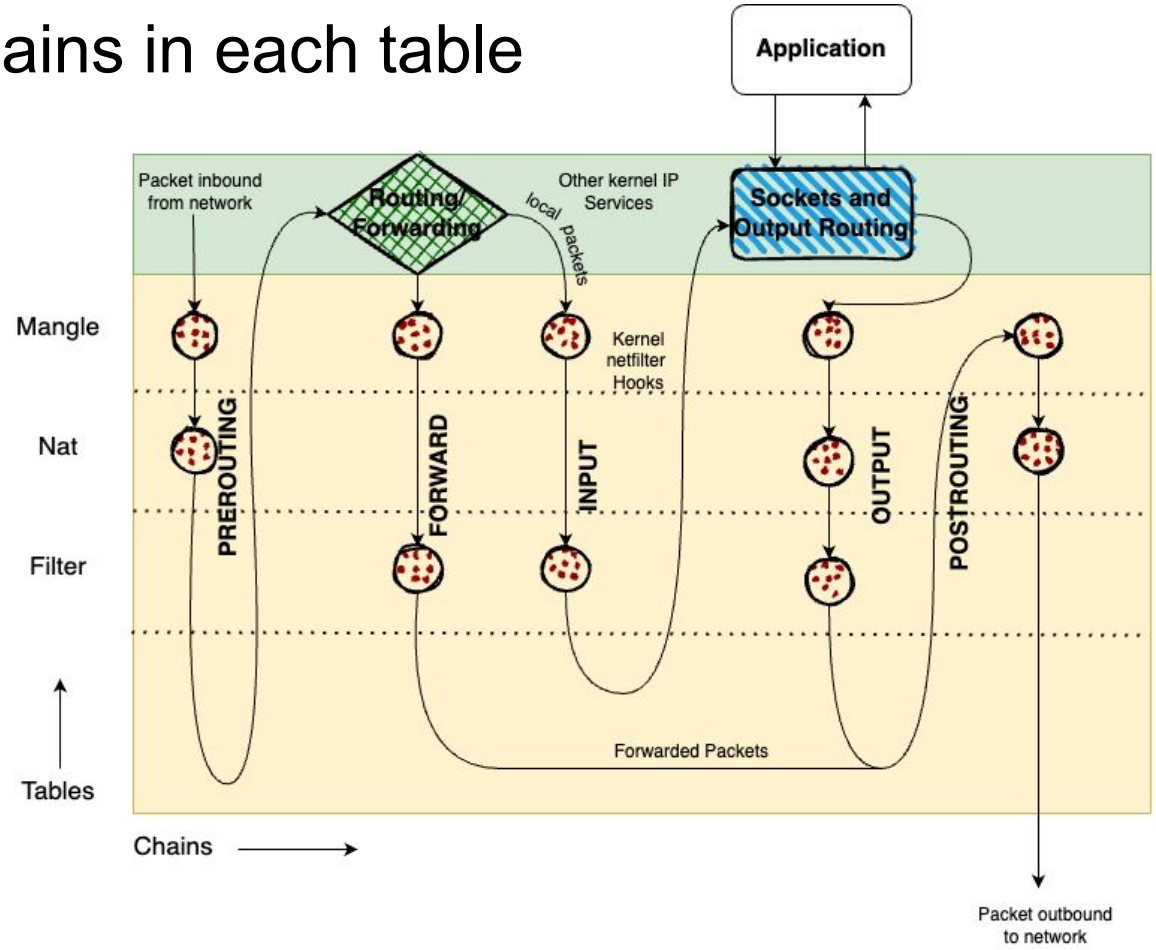
- **iptables** uses a **set of tables** containing built-in or user defined rules to accept, reject or forward packets..

iptables

- The **Filter Table** is used to control the **flow of packets** in and out of a system.
- The **NAT Table** is used to **redirect traffic** to other interfaces on the network.
- The **Mangle Table** is used to modify packet headers.



Hooks/Chains in each table



Building Blocks of iptables

- **Tables:** A table consists of several chains.
- **Chains:** String of rules.
- **Rules:** A rule is a statement that tells the system what to do with a packet. The outcome, where a packet is sent, is called a **target**.
- **Targets:** A target is a decision of what to do with a packet. Typically, this is to **accept**, **drop**, or **reject** it.

iptables commands

- **-t or –table :** Specifies the matching table
- **-A or –append chain rule :** Append rule to the end of chain
- **-L chain :** List rules in chain
- **-p** : Protocol (tcp, udp, ssh, icmp, ping)
- **-s or –source :** Specifies the source
- **-j target :** Specifies target to jump on after match
- **-F** : Flushes the rules

Syntax for iptables

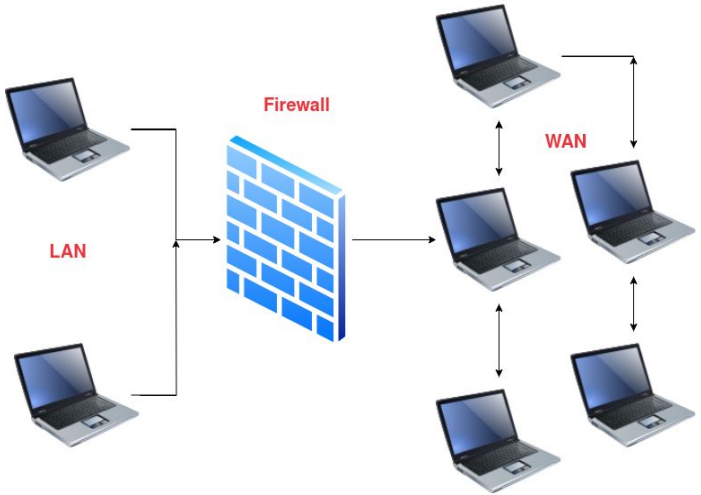
```
sudo iptables -t table_name -A
chain/hook_name filter_condition -j action
```

Firewall / Packet Filter

Firewall

- A hardware or software-based network security device, which monitors all incoming and outgoing traffic based on a defined set of security rules.

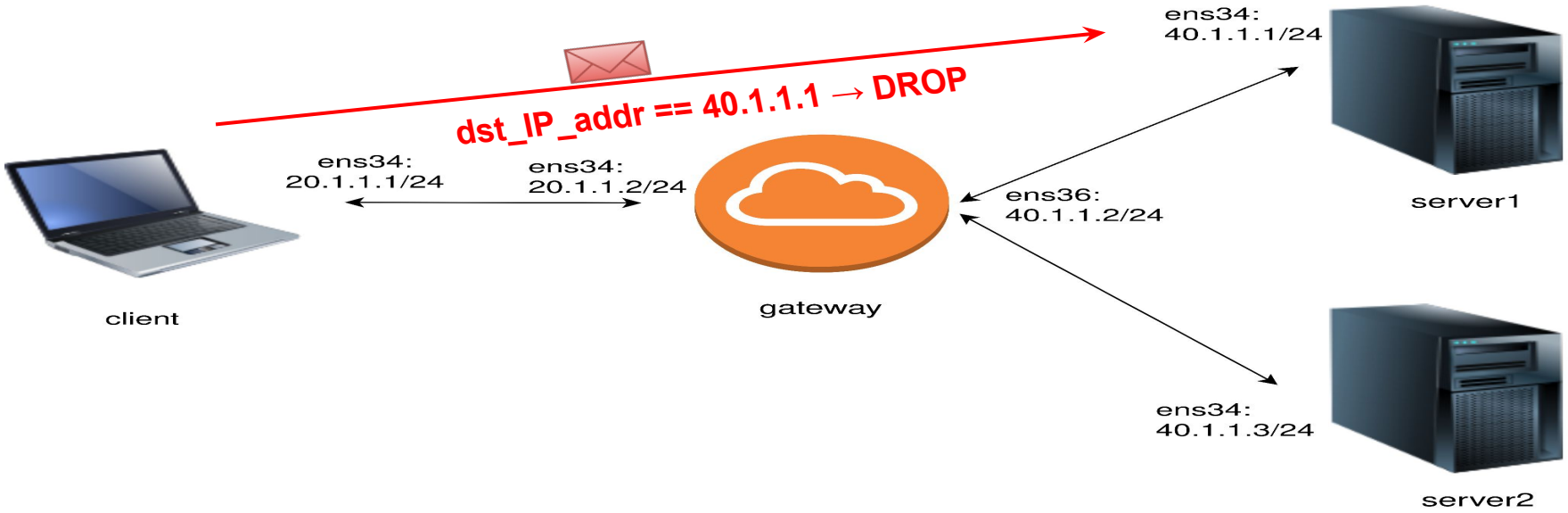
- Once the rule is matched, it **accepts**, **rejects** or **drops** that specific traffic.

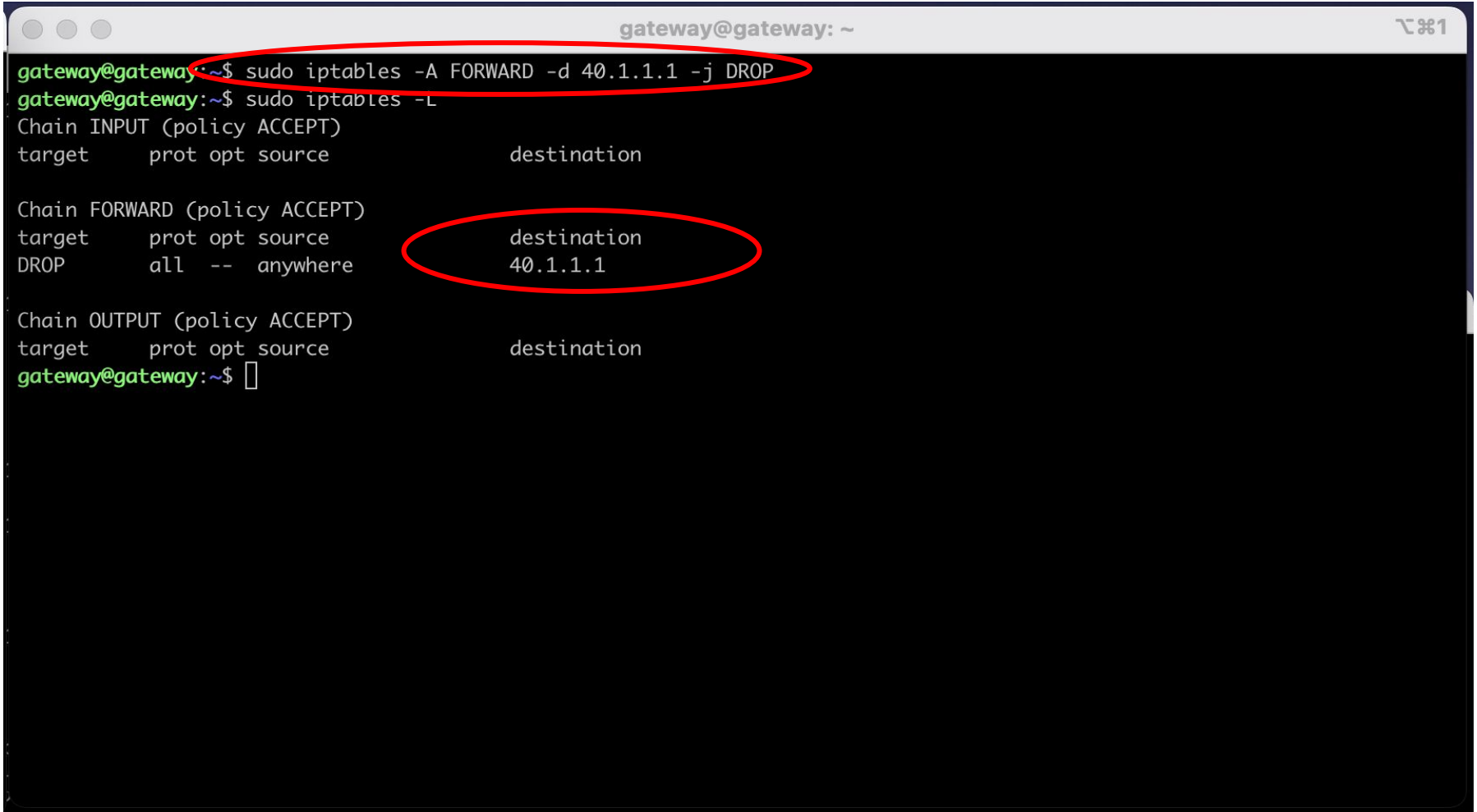


Scenario 1

To drop all the packets from client to server1

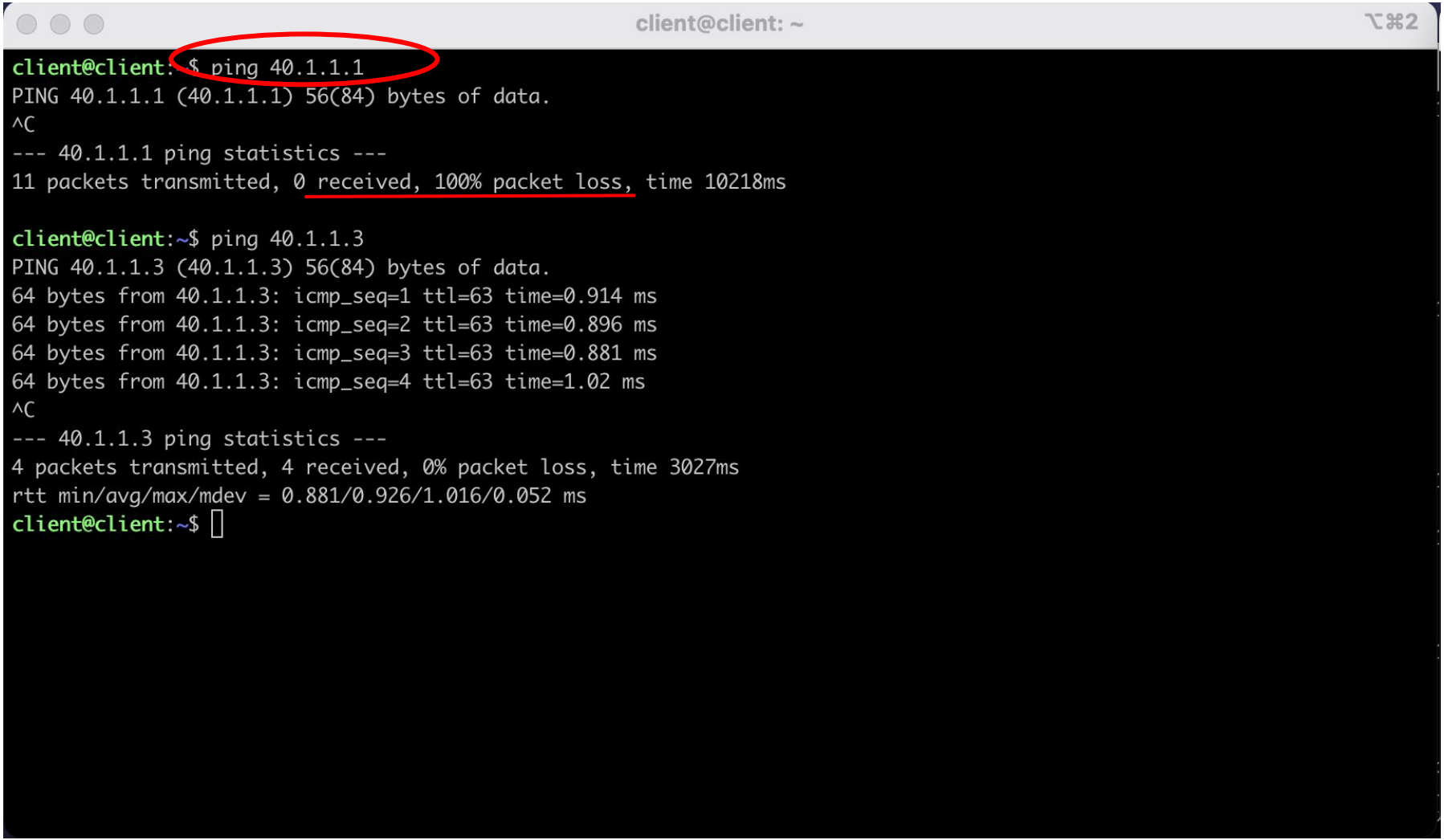
```
sudo iptables -A FORWARD -d 40.1.1.1 -j DROP
```




gateway@gateway: ~
⌥⌘1
gateway@gateway:~$ sudo iptables -A FORWARD -d 40.1.1.1 -j DROP
gateway@gateway:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target   prot opt source   destination

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

Chain OUTPUT (policy ACCEPT)
target   prot opt source   destination
gateway@gateway:~$


client@client: ~
⌥⌘2
client@client:~$ ping 40.1.1.1
PING 40.1.1.1 (40.1.1.1) 56(84) bytes of data.
^C
--- 40.1.1.1 ping statistics ---
11 packets transmitted, 0 received, 100% packet loss, time 10218ms
client@client:~$ ping 40.1.1.3
PING 40.1.1.3 (40.1.1.3) 56(84) bytes of data.
64 bytes from 40.1.1.3: icmp_seq=1 ttl=63 time=0.914 ms
64 bytes from 40.1.1.3: icmp_seq=2 ttl=63 time=0.896 ms
64 bytes from 40.1.1.3: icmp_seq=3 ttl=63 time=0.881 ms
64 bytes from 40.1.1.3: icmp_seq=4 ttl=63 time=1.02 ms
^C
--- 40.1.1.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3027ms
rtt min/avg/max/mdev = 0.881/0.926/1.016/0.052 ms
client@client:~$

netcat (nc)

- Used for
  - **network communication**
  - **port scanning**
  - **file transfer**
  - **network testing**.
- Run the following command on the device that will act as the server:

```
nc -l <port_number>
```

- To connect to this device from another device:

```
nc <host_ip_address>  <port_number>
```

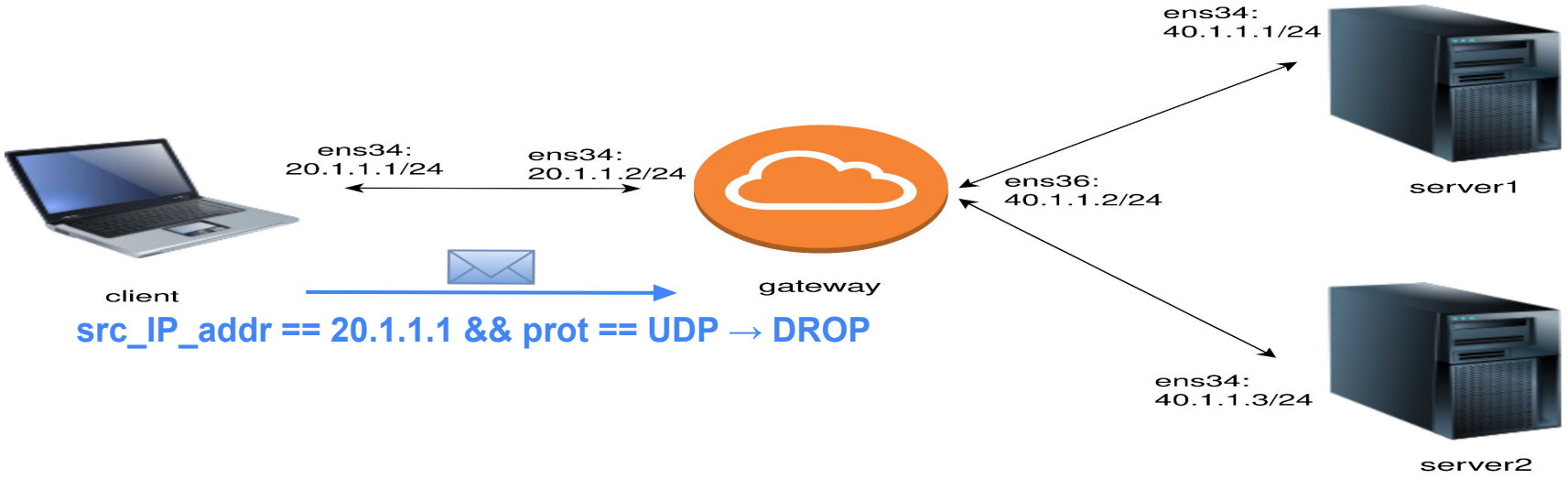

server1@server1: ~
⌥⌘3
server1@server1:~$ nc -l -p 1234
hi
we are students

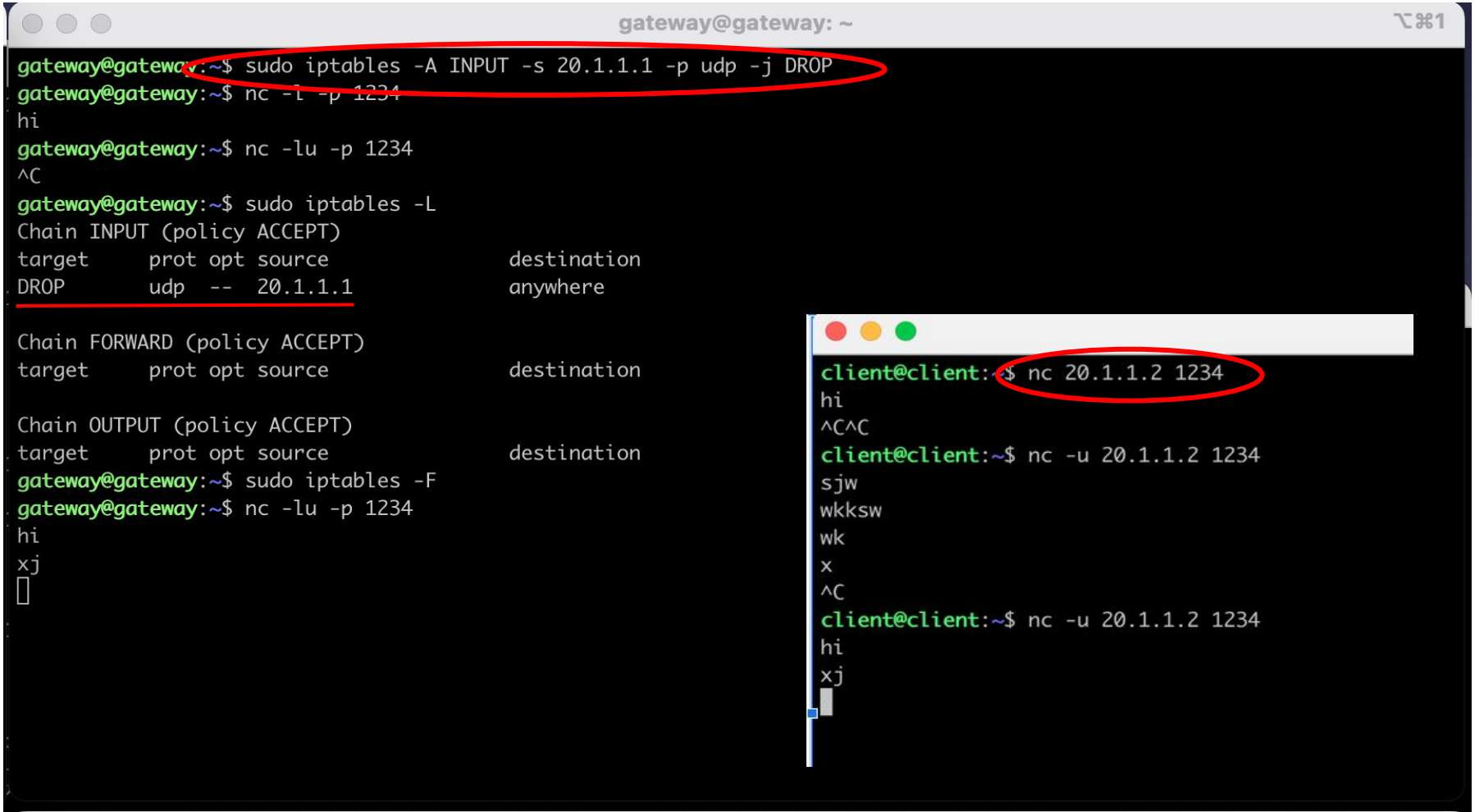

client@client: ~
⌥⌘2
client@client:~$ nc 40.1.1.1 1234
hi
we are students

Scenario 2

To block the UDP packets and allow only TCP packets from client to gateway

```
sudo iptables -A INPUT -s 20.1.1.1 -p udp -j DROP
```

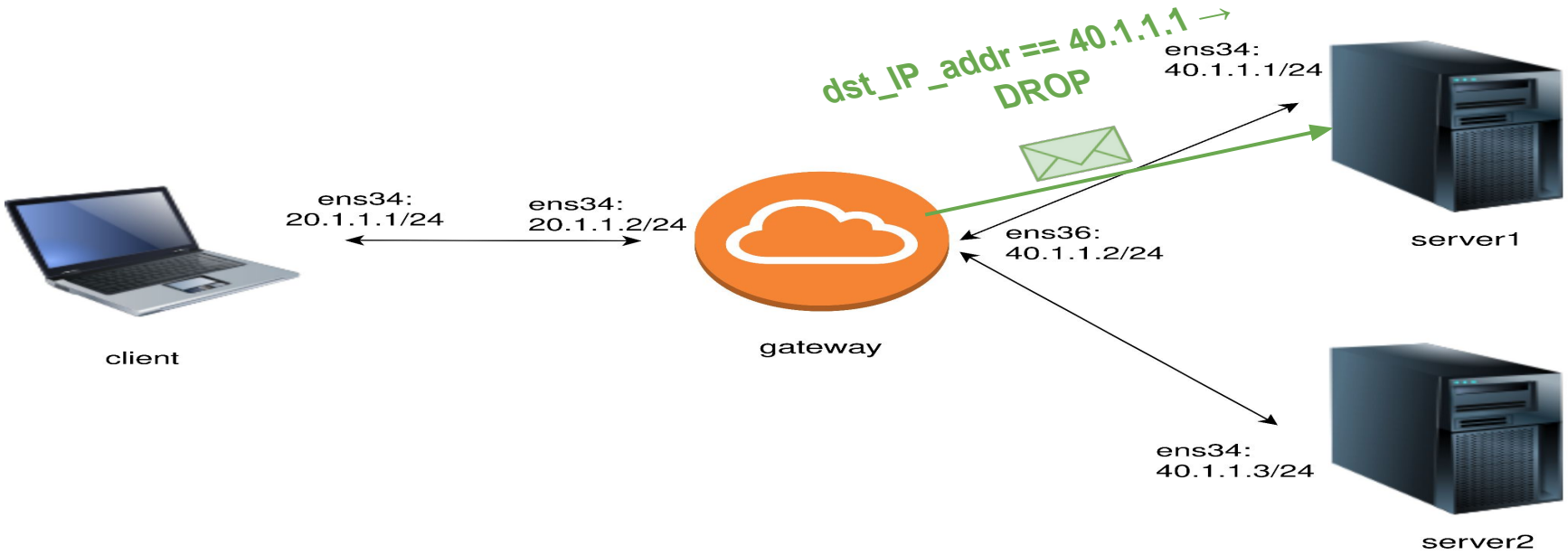


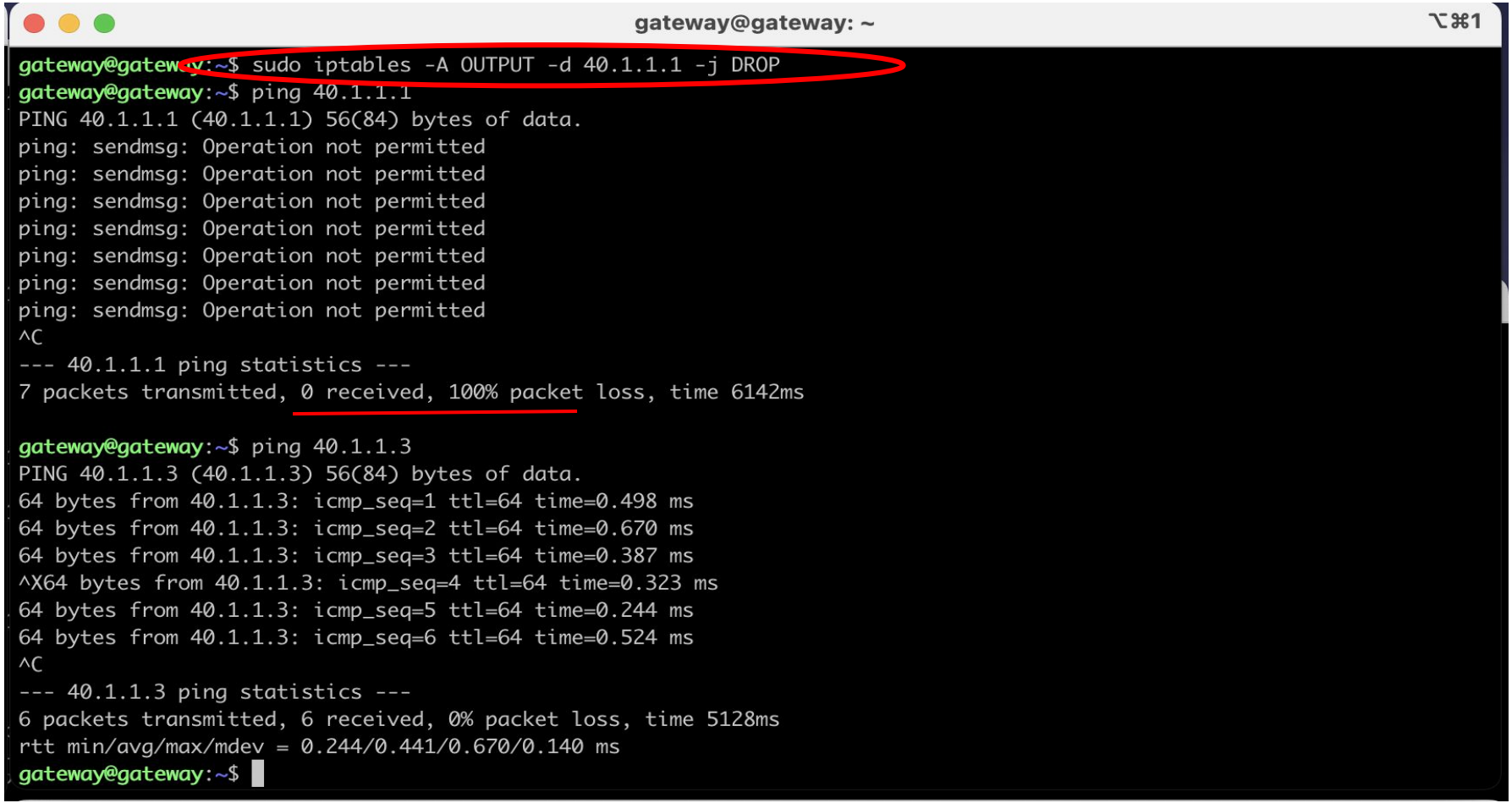

gateway@gateway: ~
gateway@gateway:~$ sudo iptables -A INPUT -s 20.1.1.1 -p udp -j DROP
gateway@gateway:~$ nc -l -p 1234
hi
gateway@gateway:~$ nc -lu -p 1234
^C
gateway@gateway:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
DROP udp -- 20.1.1.1 anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
gateway@gateway:~$ sudo iptables -F
gateway@gateway:~$ nc -lu -p 1234
hi
xj
client@client:~$ nc 20.1.1.2 1234
hi
^C^C
client@client:~$ nc -u 20.1.1.2 1234
sjw
wkksw
wk
x
^C
client@client:~$ nc -u 20.1.1.2 1234
hi
xj

Scenario 3

To block all the packets from gateway to server1

```
sudo iptables -A OUTPUT -d 40.1.1.1 -j  DROP
```




gateway@gateway: ~
⌥⌘1
gateway@gateway:~$ sudo iptables -A OUTPUT -d 40.1.1.1 -j DROP
gateway@gateway:~$ ping 40.1.1.1
PING 40.1.1.1 (40.1.1.1) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 40.1.1.1 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6142ms
gateway@gateway:~$ ping 40.1.1.3
PING 40.1.1.3 (40.1.1.3) 56(84) bytes of data.
64 bytes from 40.1.1.3: icmp_seq=1 ttl=64 time=0.498 ms
64 bytes from 40.1.1.3: icmp_seq=2 ttl=64 time=0.670 ms
64 bytes from 40.1.1.3: icmp_seq=3 ttl=64 time=0.387 ms
^X64 bytes from 40.1.1.3: icmp_seq=4 ttl=64 time=0.323 ms
64 bytes from 40.1.1.3: icmp_seq=5 ttl=64 time=0.244 ms
64 bytes from 40.1.1.3: icmp_seq=6 ttl=64 time=0.524 ms
^C
--- 40.1.1.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5128ms
rtt min/avg/max/mdev = 0.244/0.441/0.670/0.140 ms
gateway@gateway:~$

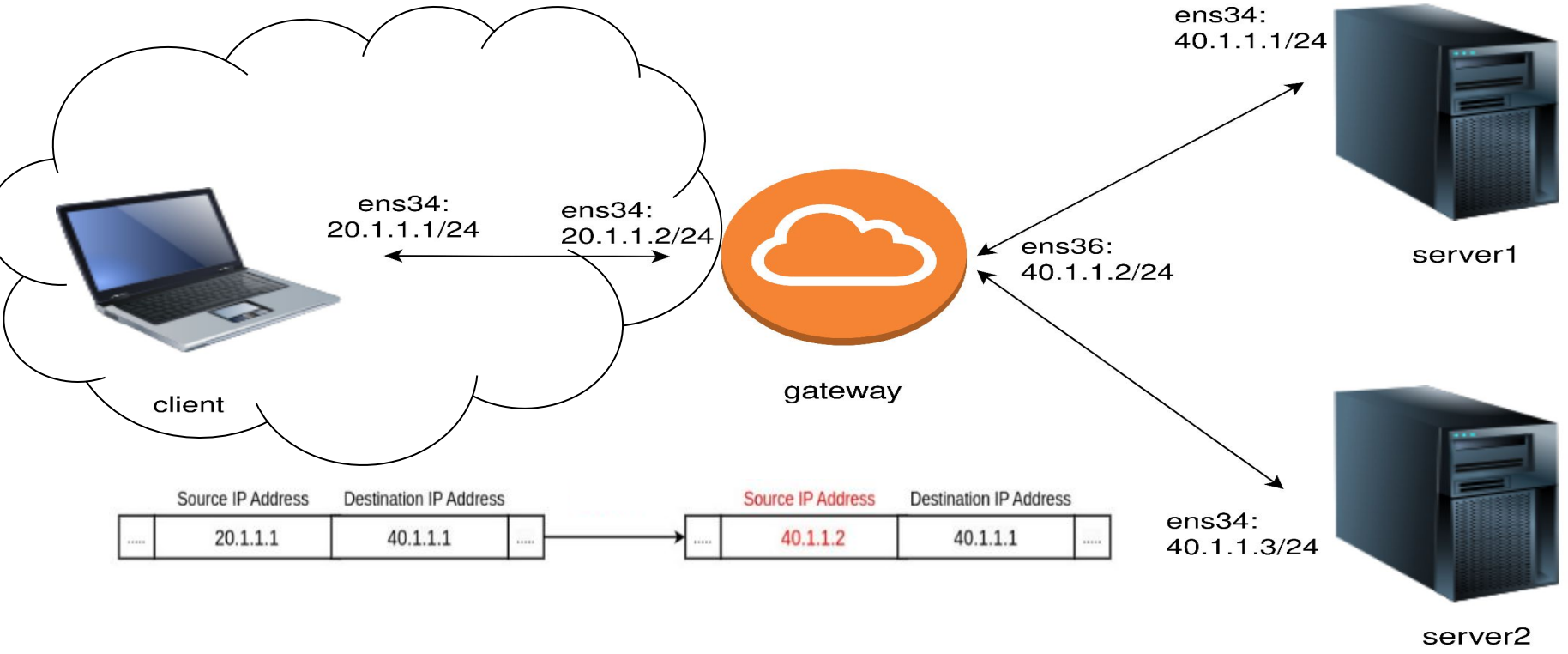
NAT (Network Address Translation)

NAT

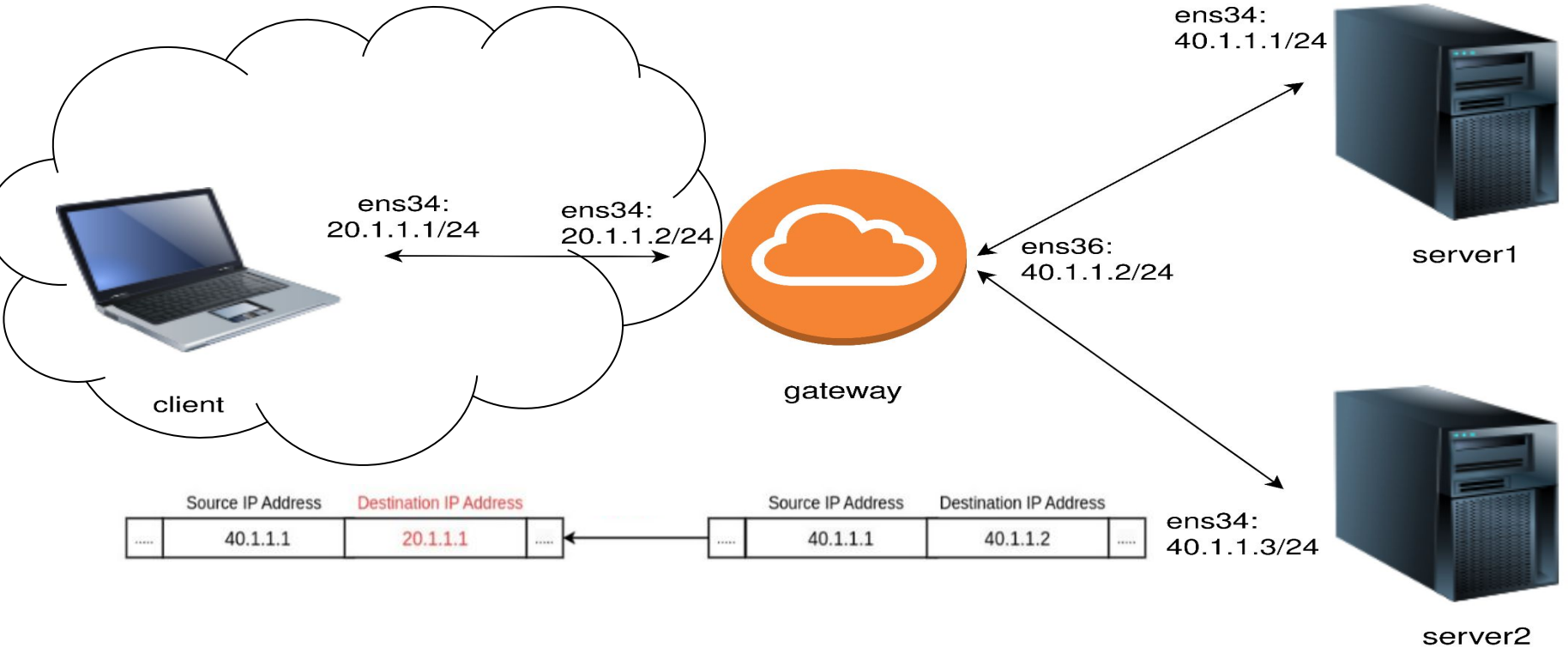
- Network Address Translation (NAT) is used to convert a private IP address into a public IP address and vice versa.

- The range of private addresses known as RFC 1918 are:
    - Class A 10.0.0.0 - 10.255.255.255
    - Class B 172.16.0.0 - 172.31.255.255
    - Class C 192.168.0.0 - 192.168.255.255

Objective 1: Private IP address —> Public IP address

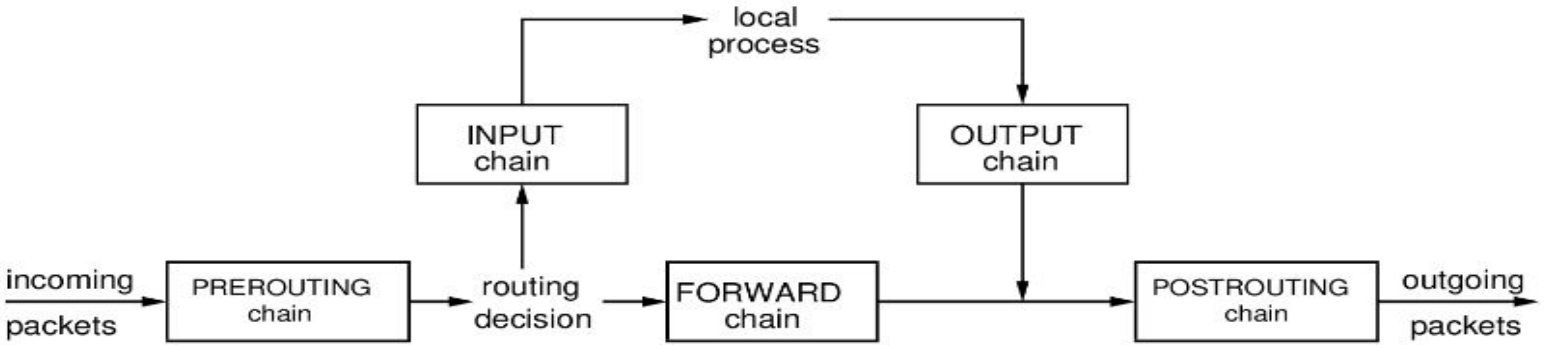


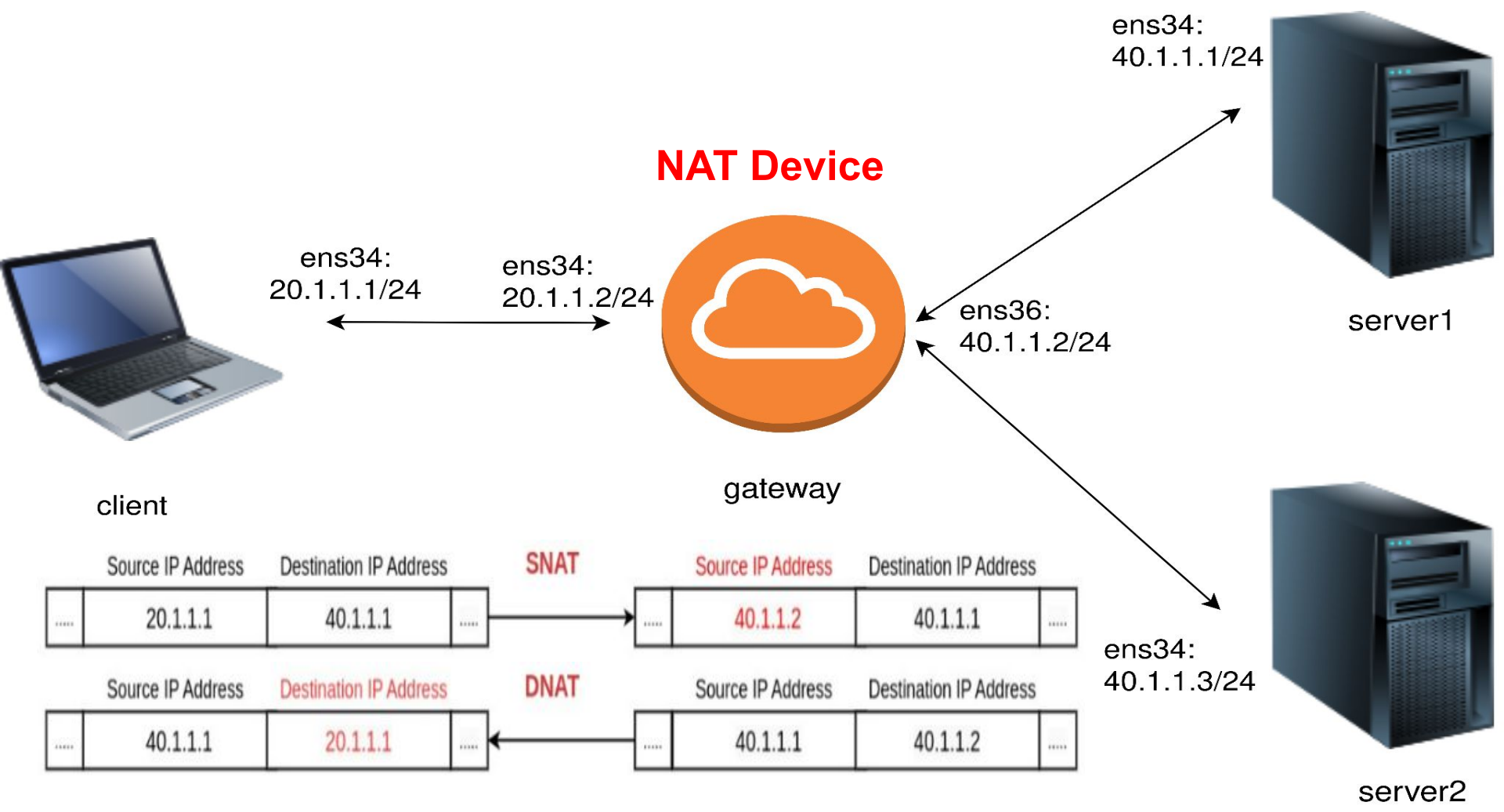
Objective 2: Public IP address —> Private IP address



Types of NAT

- **SNAT:** Used as a target in NAT table, within the POSTROUTING chain.
- **DNAT:** Used as a target NAT table, within the PREROUTING chain.



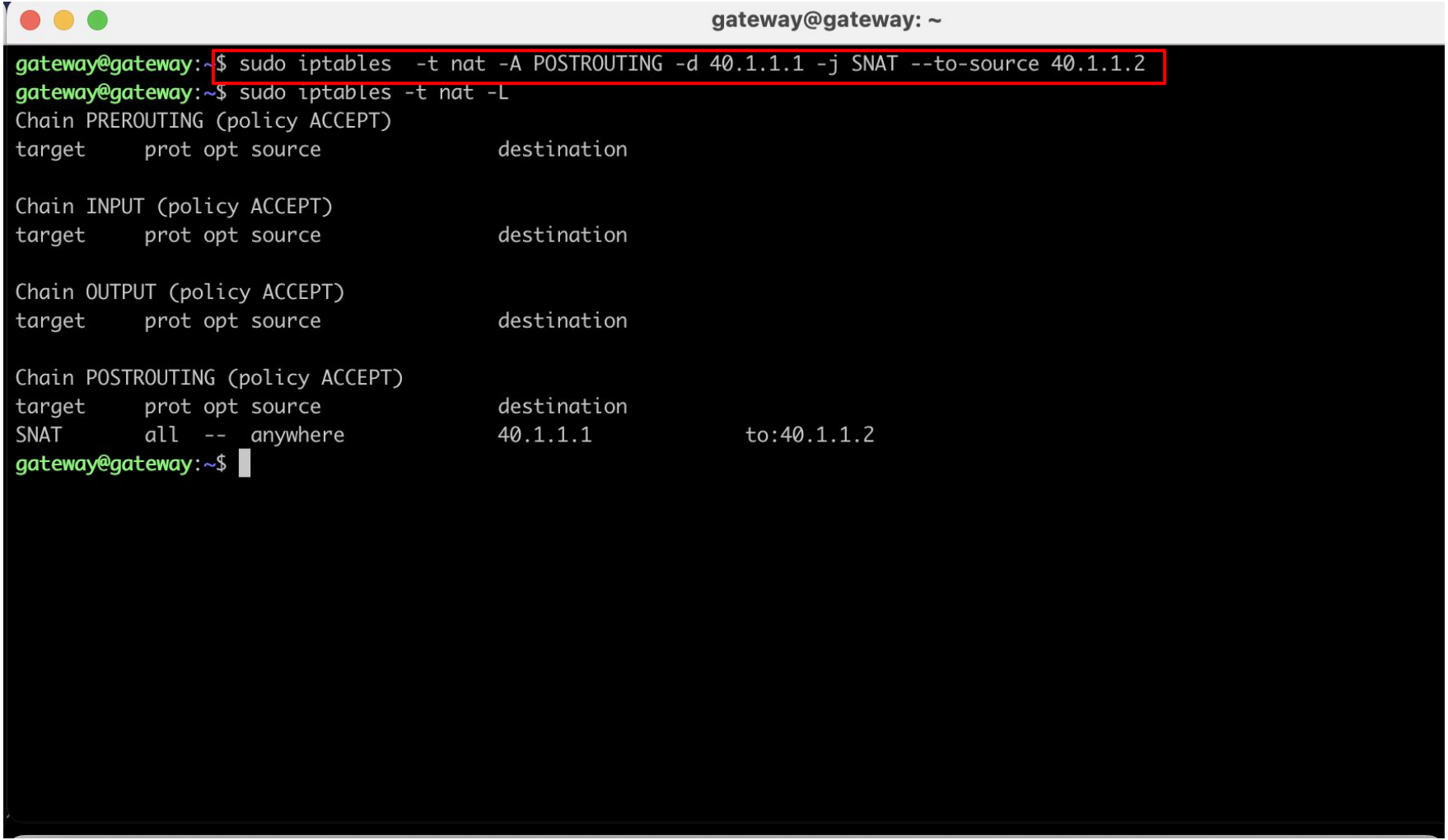

NAT Device
ens34:
20.1.1.1/24
ens34:
20.1.1.2/24
ens34:
40.1.1.1/24
ens36:
40.1.1.2/24
ens34:
40.1.1.3/24
client
gateway
server1
server2
Source IP Address
Destination IP Address
SNAT
20.1.1.1
40.1.1.1
40.1.1.2
40.1.1.1
DNAT
40.1.1.1
20.1.1.1
40.1.1.1
40.1.1.2

SNAT (Source Network Address Translation)

SNAT using NAT Table

```
sudo iptables -t nat -A POSTROUTING -d 40.1.1.1 -j SNAT -- to-source 40.1.1.2
```

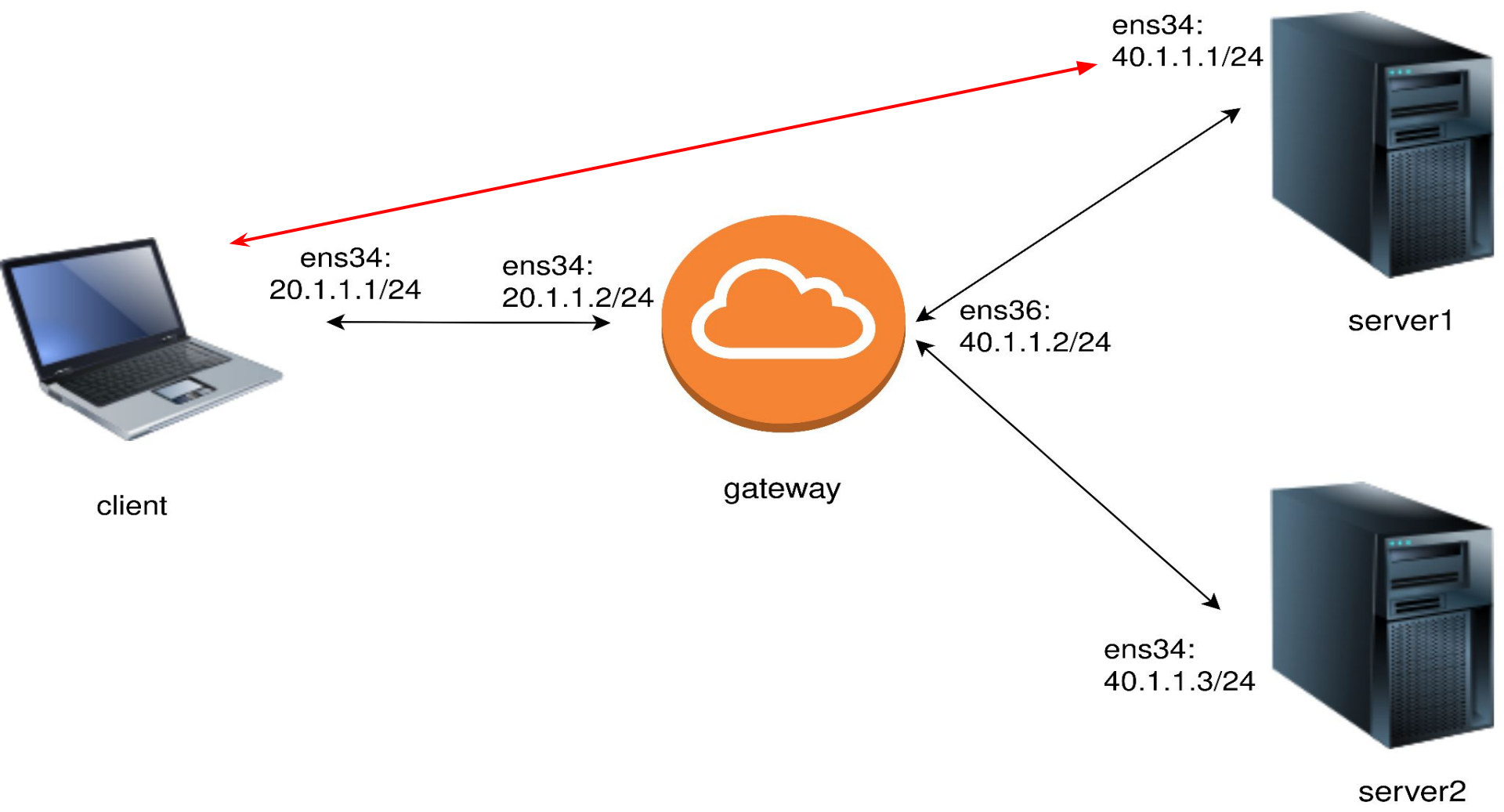
```
sudo iptables \
-t nat \                    # in the nat table
-A POSTROUTING \            # Append a rule to the POSTROUTING chain
  -d 40.1.1.1 \             #   if a packet is sent to the destination with IP 40.1.1.1
-j SNAT \                   #   then use the SNAT target
--to-source 40.1.1.2        #    and change the SRC IP to 40.1.1.2
```


gateway@gateway: ~
gateway@gateway:~$ sudo iptables -t nat -A POSTROUTING -d 40.1.1.1 -j SNAT --to-source 40.1.1.2
gateway@gateway:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target   prot opt source   destination

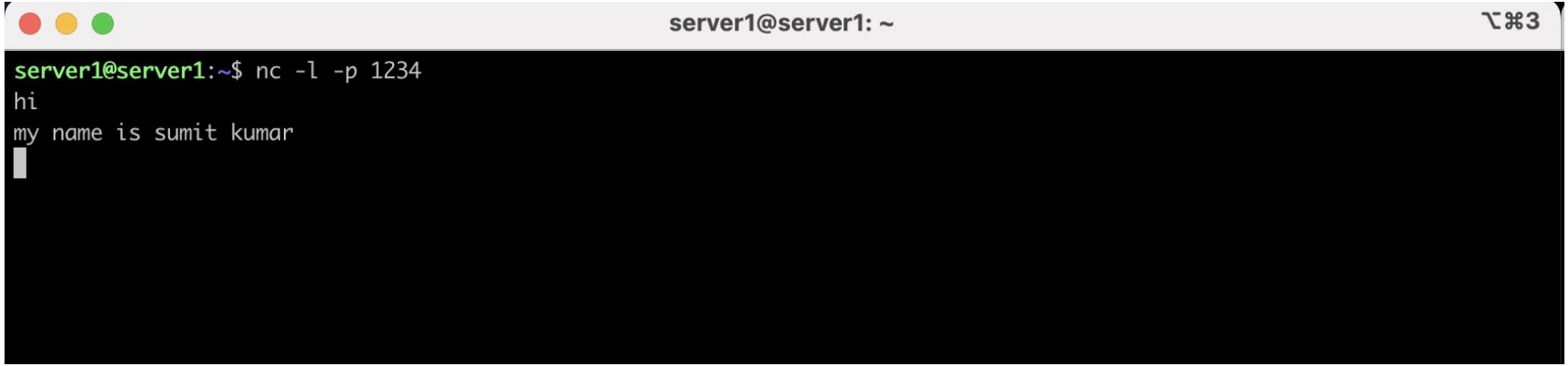
Chain INPUT (policy ACCEPT)
target   prot opt source   destination

Chain OUTPUT (policy ACCEPT)
target   prot opt source   destination

Chain POSTROUTING (policy ACCEPT)
target   prot opt source   destination
SNAT   all -- anywhere   40.1.1.1   to:40.1.1.2
gateway@gateway:~$

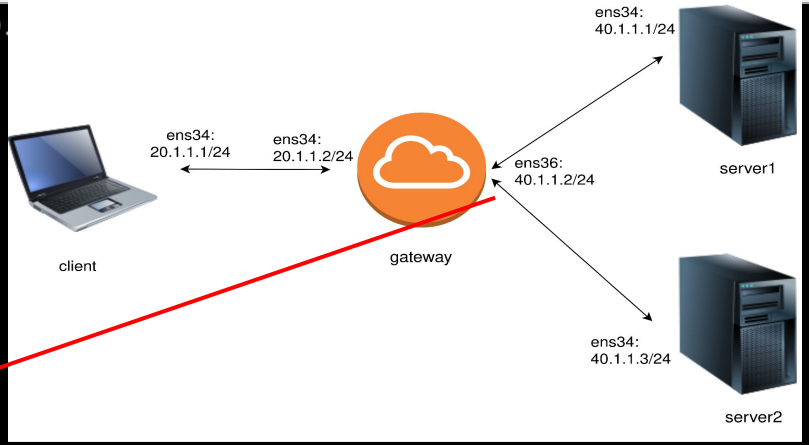

ens34:
40.1.1.1/24
server1
ens34:
20.1.1.1/24
ens34:
20.1.1.2/24
ens36:
40.1.1.2/24
client
gateway
ens34:
40.1.1.3/24
server2


client@client: ~
⌥⌘2
client@client:~$ nc 40.1.1.1 1234
hi
my name is sumit kumar


server1@server1: ~
⌥⌘3
server1@server1:~$ nc -l -p 1234
hi
my name is sumit kumar

gateway@gateway: ~

```
gateway@gateway:~$ sudo iptables  -t nat -A POSTROUTING -d 40.
gateway@gateway:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (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
SNAT       all  --  anywhere             40.1.1.1             to:40.1.1.2
gateway@gateway:~$ sudo tshark -i ens36
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens36'
    1 0.000000000     40.1.1.2 → 40.1.1.1     TCP 69 57250 → 1234 [PSH, ACK] Seq=1 Ack=1 Win=502 Len=3 TSval=338984861 TSe
    2 0.000555822     40.1.1.1 → 40.1.1.2     TCP 66 1234 → 57250 [ACK] Seq=1 Ack=4 Win=510 Len=0 TSval=3860445448 TSecr=3
    3 5.239607559 VMware_f5:64:07 → VMware_ef:6c:9e ARP 60 Who has 40.1.1.2? Tell 40.1.1.1
    4 5.239628694 VMware_ef:6c:9e → VMware_f5:64:07 ARP 42 40.1.1.2 is at 00:0c:29:ef:6c:9e
    5 5.240980979 VMware_ef:6c:9e → VMware_f5:64:07 ARP 42 Who has 40.1.1.1? Tell 40.1.1.2
    6 5.241231339 VMware_f5:64:07 → VMware_ef:6c:9e ARP 60 40.1.1.1 is at 00:0c:29:f5:64:07
    7 12.841885134     40.1.1.2 → 40.1.1.1     TCP 89 57250 → 1234 [PSH, ACK] Seq=4 Ack=1 Win=502 Len=23 TSval=338997703 T
    8 12.842250459     40.1.1.1 → 40.1.1.2     TCP 66 1234 → 57250 [ACK] Seq=1 Ack=27 Win=510 Len=0 TSval=3860458289 TSecr
```

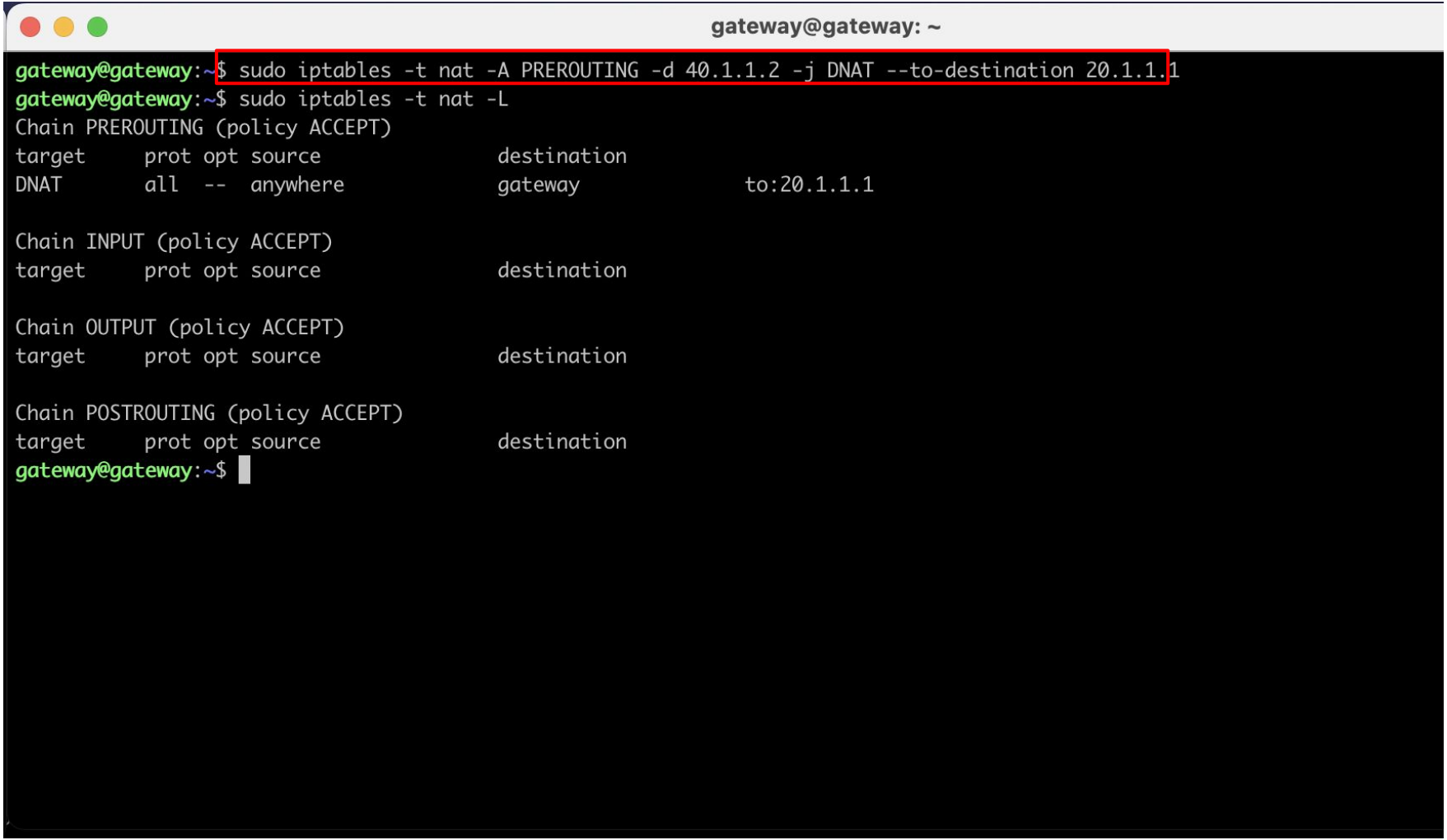


DNAT(Destination Network Address Translation)

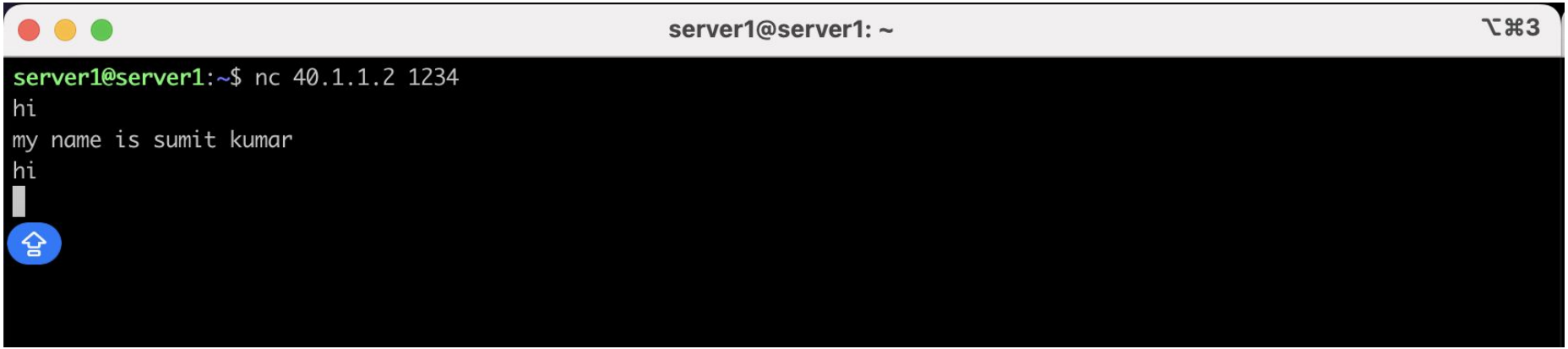
DNAT using NAT Table

```
sudo iptables -t nat -A PREROUTING -d 40.1.1.2 -j DNAT - - to-destination 20.1.1.1
```

```
sudo iptables \
-t nat \                        # in the nat table
-A POSTROUTING \                # Append a rule to the POSTROUTING chain
-d 40.1.1.2 \                   #   if a packet is sent to the destination with IP 40.1.1.2
-j DNAT \                       #   then use the DNAT target
--to-destination 20.0.0.1 #     and change the SRC IP to 20.0.0.1
```

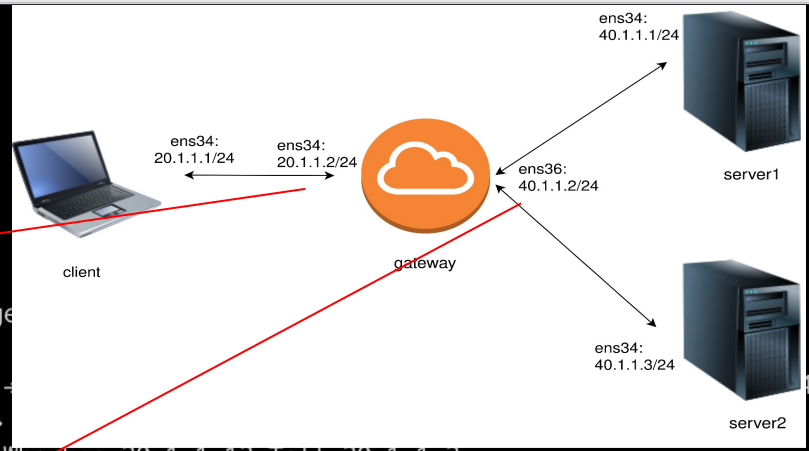

gateway@gateway: ~
gateway@gateway:~$ sudo iptables -t nat -A PREROUTING -d 40.1.1.2 -j DNAT --to-destination 20.1.1.1
gateway@gateway:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
DNAT all -- anywhere gateway to:20.1.1.1
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
Chain POSTROUTING (policy ACCEPT)
target prot opt source destination
gateway@gateway:~$


client@client: ~
⌥⌘2
client@client:~$ nc -l -p 1234
hi
my name is sumit kumar
hi


server1@server1: ~
⌥⌘3
server1@server1:~$ nc 40.1.1.2 1234
hi
my name is sumit kumar
hi

gateway@gateway: ~

```
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
gateway@gateway:~$ sudo tshark -i ens34
Running as user "root" and group "root". This could be dange
Capturing on 'ens34'
    1 0.000000000     40.1.1.1 → 20.1.1.1     TCP 69 43074 →                                  416298 TS
    2 0.000216890     20.1.1.1 → 40.1.1.1     TCP 66 1234 →                                   TSecr=38
    3 5.152708884 VMware_ef:6c:8a → VMware_b8:2b:9b ARP 42 Who has 20.1.1.1? Tell 20.1.1.2
    4 5.152966944 VMware_b8:2b:9b → VMware_ef:6c:8a ARP 60 20.1.1.1 is at 00:0c:29:b8:2b:9b
    5 5.243106943 VMware_b8:2b:9b → VMware_ef:6c:8a ARP 60 Who has 20.1.1.2? Tell 20.1.1.1
    6 5.243119983 VMware_ef:6c:8a → VMware_b8:2b:9b ARP 42 20.1.1.2 is at 00:0c:29:ef:6c:8a
^C6 packets captured
gateway@gateway:~$ sudo tshark -i ens36
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens36'
    1 0.000000000     40.1.1.1 → 40.1.1.2     TCP 70 43074 → 1234 [PSH, ACK] Seq=1 Ack=1 Win=502 Len=4 TSval=3861904550 TS
    2 0.000236439     40.1.1.2 → 40.1.1.1     TCP 66 1234 → 43074 [ACK] Seq=1 Ack=5 Win=510 Len=0 TSval=340443953 TSecr=38
    3 5.080410692 VMware_f5:64:07 → VMware_ef:6c:9e ARP 60 Who has 40.1.1.2? Tell 40.1.1.1
    4 5.080426100 VMware_ef:6c:9e → VMware_f5:64:07 ARP 42 40.1.1.2 is at 00:0c:29:ef:6c:9e
    5 5.095399013 VMware_ef:6c:9e → VMware_f5:64:07 ARP 42 Who has 40.1.1.1? Tell 40.1.1.2
    6 5.095610934 VMware_f5:64:07 → VMware_ef:6c:9e ARP 60 40.1.1.1 is at 00:0c:29:f5:64:07
```

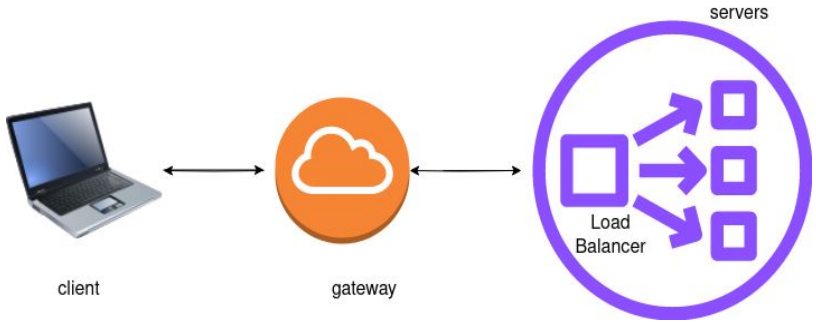


Load Balancing

Load Balancing

- **Load balancing** distributes incoming traffic across servers.

- **iptables** includes a module called **statistic** that can skip or accept rule based on some condition.
- **DNAT** target of iptables can be used with the **statistic** module to perform load balancing



..contd.

The **statistic module** support two different modes:

random: based on probability

nth: based on round robin algorithm

Note: LB occurs only at the connection phase of the TCP protocol. Once the connection has been established, the connection will always be routed to the same server.

Random Balancing

- **Based on probability**

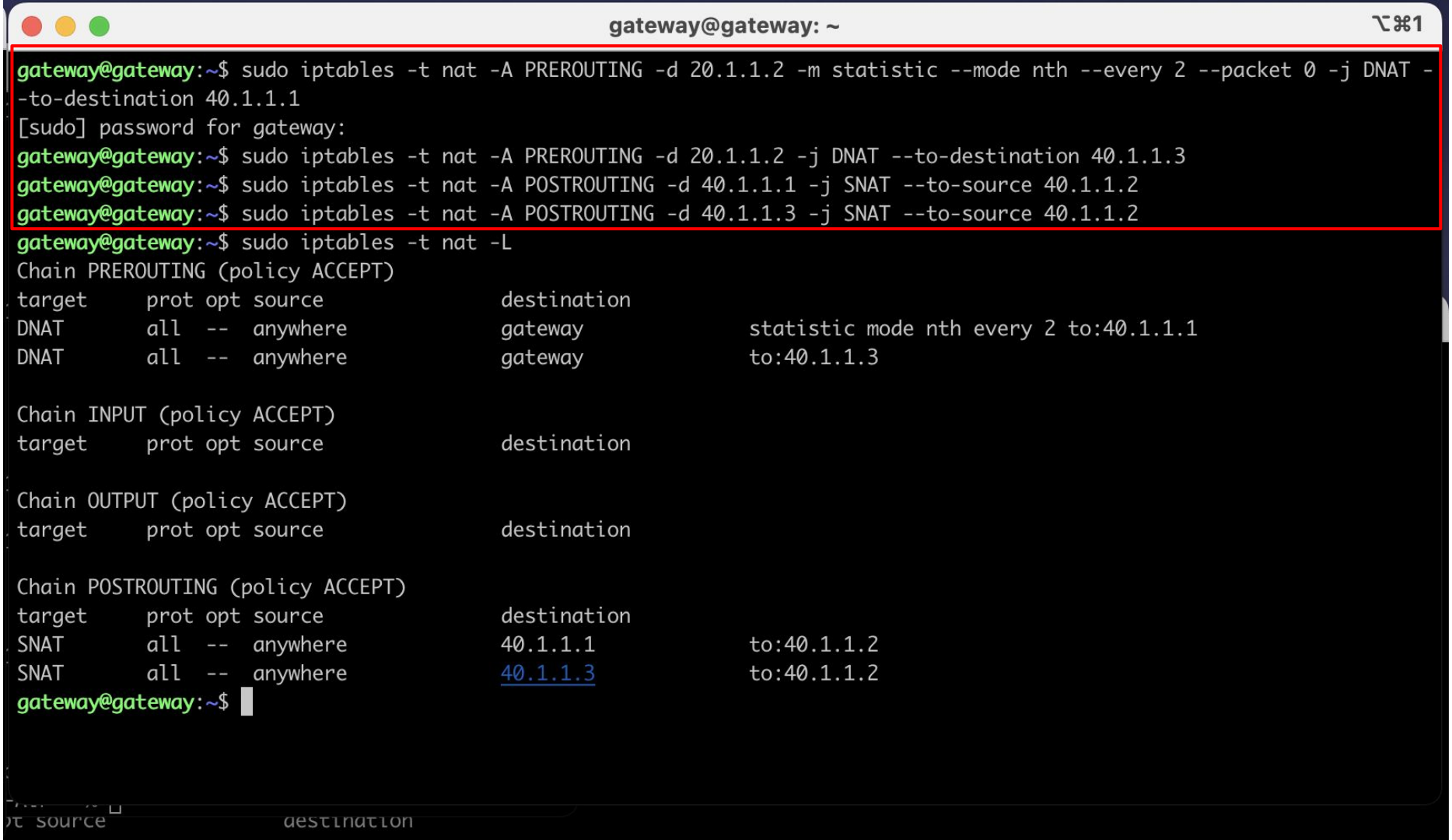
```
iptables -A PREROUTING -t nat -d 20.1.1.2 \
    -m statistic --mode random --probability 0.5 \
    -j DNAT --to-destination 40.1.1.1

iptables -A PREROUTING -t nat -p tcp \
    -d 20.1.1.2 -j DNAT --to-destination 40.1.1.3
```

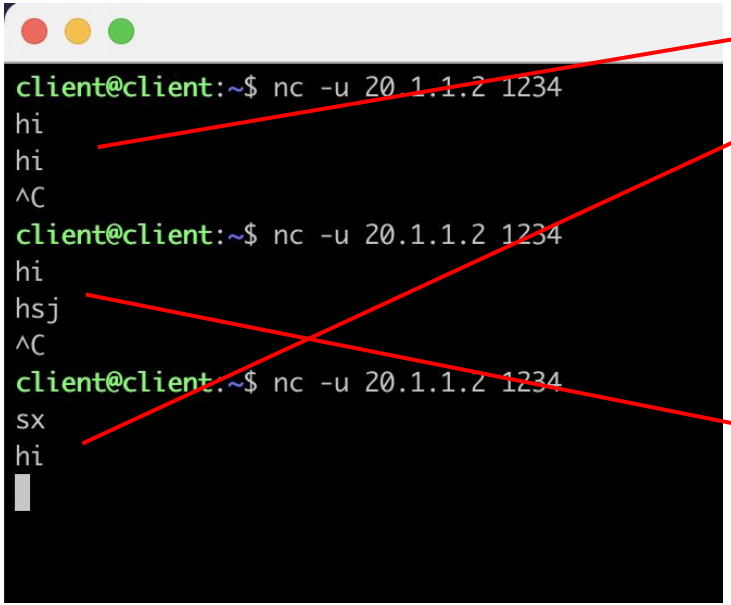
Round Robin Balancing

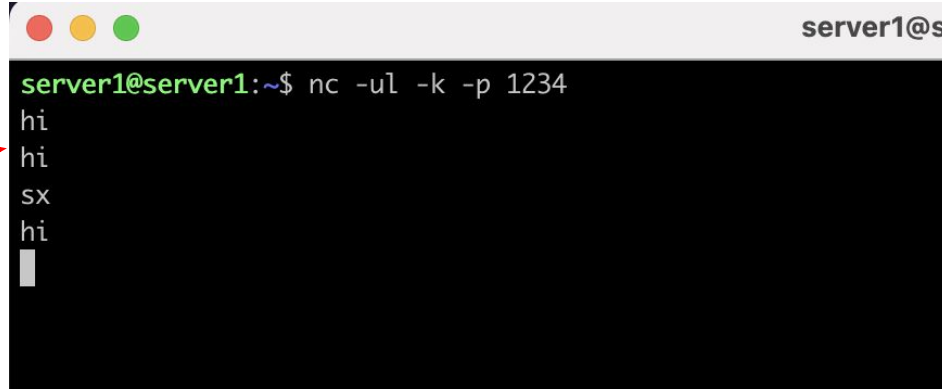
The algorithm takes two different parameters **every(n)** and **packet(p)**. The rule will be evaluated every **n packets** after packet **p**

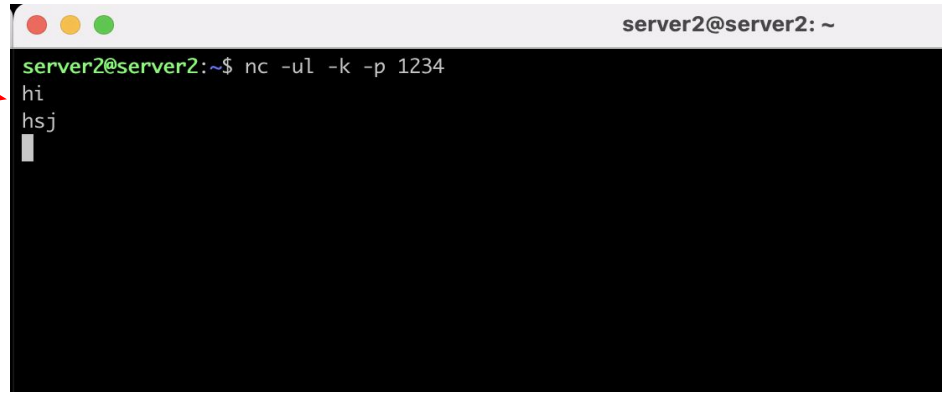
```
sudo iptables -t nat -A PREROUTING -d 20.1.1.2
        -m statistic --mode nth --every 2 --packet 0
        -j DNAT --to-destination 40.1.1.1
```

```
sudo iptables -t nat -A PREROUTING \
        -d 20.1.1.2 -j DNAT --to-destination 40.1.1.3
```


gateway@gateway: ~
⌥⌘1
gateway@gateway:~$ sudo iptables -t nat -A PREROUTING -d 20.1.1.2 -m statistic --mode nth --every 2 --packet 0 -j DNAT --to-destination 40.1.1.1
[sudo] password for gateway:
gateway@gateway:~$ sudo iptables -t nat -A PREROUTING -d 20.1.1.2 -j DNAT --to-destination 40.1.1.3
gateway@gateway:~$ sudo iptables -t nat -A POSTROUTING -d 40.1.1.1 -j SNAT --to-source 40.1.1.2
gateway@gateway:~$ sudo iptables -t nat -A POSTROUTING -d 40.1.1.3 -j SNAT --to-source 40.1.1.2
gateway@gateway:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
DNAT all -- anywhere gateway statistic mode nth every 2 to:40.1.1.1
DNAT all -- anywhere gateway to:40.1.1.3
Chain INPUT (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
SNAT all -- anywhere 40.1.1.1 to:40.1.1.2
SNAT all -- anywhere 40.1.1.3 to:40.1.1.2
gateway@gateway:~$


client@client:~$ nc -u 20.1.1.2 1234
hi
hi
^C
client@client:~$ nc -u 20.1.1.2 1234
hi
hsj
^C
client@client:~$ nc -u 20.1.1.2 1234
sx
hi


server1@s
server1@server1:~$ nc -ul -k -p 1234
hi
hi
sx
hi


server2@server2: ~
server2@server2:~$ nc -ul -k -p 1234
hi
hsj

Reference

- Ifconfig, Nestat, Ping, traceroute
- Iperf, tc
- Netfilter
- iptables
- https://scalingo.com/blog/iptables
- https://docs.kernel.org/networking/nf_flowtable.html


Thank
you

Scan the QR code to attempt the MCQ